SISTEM PENDETEKSIAN PENYUSUPAN JARINGAN KOMPUTER DENGAN *ACTIVE RESPONSE* MENGGUNAKAN METODE *HYBRID INTRUSION DETECTION, SIGNATURES* DAN *ANOMALY DETECTION*

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik
Pada Jurusan Teknik Informatika

oleh :

WENNI SYAFITRI
10551001512

FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS ISLAM NEGERI SULTAN SYARIF KASIM RIAU
PEKANBARU
2010

SISTEM PENDETEKSIAN PENYUSUPAN JARINGAN KOMPUTER DENGAN *ACTIVE RESPONSE* MENGGUNAKAN METODE *HYBRID INTRUSION DETECTION, SIGNATURES* DAN *ANOMALY DETECTION*

WENNI SYAFITRI
10551001512

Tanggal Sidang : 28 Januari 2010
Periode Wisuda : Februari 2010

Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Islam Negeri Sultan Syarif Kasim Riau

ABSTRAK

Kemajuan teknologi internet menyebabkan meningkatnya kebutuhan akan keamanan data. Terjadinya perkembangan *tools* yang mampu melakukan penyusupan juga mempengaruhi kebutuhan tersebut. Metode implementasi Sistem Pendeteksian Penyusupan Jaringan *(IDS)* dan metode penganalisaan penyusupan memiliki kelebihan dan kekurangan yang dapat saling melengkapi. Ada banyak *IDS* pada masa sekarang, namun hanya sebuah *IDS* yang berbasiskan *open source* yaitu *snort*. Di lain hal, metode implementasi *snort* hanya terbatas *network based.*

Sistem pada Tugas Akhir ini, menggunakan metode *Hybrid Intrusion Detection System, Signatures* dan *Anomaly Detection.* Indikator yang digunakan dalam melakukan pendeteksian penyusupan yaitu *Ip Address* dan *Port Number.* Sistem ini menggunakan protokol *TCP,UDP* dan *ICMP*. Sistem ini juga dilengkapi dengan aktif respon, berupa *blocking access* terhadap penyusup.

Sistem ini diimplementasikan dengan bahasa pemrograman *Java* pada bagian *engine* unjuk kerja system dan *Java Server Pages (JSP)* pada pembangunan antarmuka pengguna, serta *database* yang digunakan yaitu *MySQL.*

Pengujian pengembangan perangkat ini dilaksanakan terdiri ada dua bagian yaitu pengujian *Link* sistem dan pengujian penyusupan jaringan. Pengujian *Link* dapat memperlihatkan terhubungnya masing-masing tampilan. Pengujian Penyusupan jaringan dilaksanakan pada kondisi pendeteksian *host* menggunakan *tools DOSHttp* dan kondisi *network* menggunakan *script Ping of Death.* Pengujian yang telah dilakukan menghasilkan kesimpulan bahwa dapat dilakukan pendeteksian, penganalisaan dan respon aktif terhadap penyusupan

Kata Kunci : *Hybrid Intrusion Detection System*; *Java; Java Server Pages (JSP);* Kebutuhan Keamanan Data; protocol *TCP,UDP*, dan *ICMP*; *Signature* dan *Anomaly Detection*; Sistem Pendeteksian Penyusupan Jaringan (*IDS)*.

INTRUSION DETECTION SYSTEM WITH ACTIVE RESPONSE USING HYBRID INTRUSION DETECTION, SIGNATURES AND ANOMALY DETECTION METHODS

WENNI SYAFITRI
10551001512

Final Exam Date : January 28th, 2010
Graduation Ceremony Period : February 2010

Informatics Engineering Departement
Faculty of Sciences and Technology
State Islamic University of Sultan Syarif Kasim Riau

ABSTRACT

The progress of internet technology increase the need of security data. The progress of tools which have intrusion ability, also influence these needed. The methods of Intrusion Detection System (IDS) implementation and methods of analyze intrusion have excess and lack, which mutually completes. There are a lot of IDS now, but just an IDS open source based is snort. Method of snort implementation is network based restricted.

This Final Task's system used Hybrid Intrusion Detection System, Signatures and Anomaly Detection Methods. The indicator which used to detect intrusion are IP Address and Port Number. This system use TCP, UDP and ICMP protocols. This system also, is completed by active response, like blocking access for intruder.

This System Implementation with Java Programming Language for engine perform and Java Server Pages (JSP) to develop user interface, The database which used is MYSQL.

There are two of development test; Link system test and intrusion test. The link system test show the connect each interface. Intrusion is executed by host detection which used DoSHttp tools and network detection which used Ping of Death's scripts. The intrusion testing conclusions are; can be detected, analyze and active response for intrusion.

.

Keywords : *Hybrid Intrusion Detection System; Intrusion Detection System (IDS); Java; Java Server Pages (JSP); Signature and Anomaly Detection; TCP, UDP and ICMP protocols; The need of data security*

DAFTAR ISI

Halaman

BAB I
PENDAHULUAN

1.1 Latar Belakang

Internet merupakan hal yang lumrah pada zaman sekarang ini. Banyak aktivitas yang dapat dilakukan dengan media internet. Dari sekedar melakukan pencarian suatu artikel atau jurnal sampai kepada konsep berbelanja di dunia maya (*e-commerce*). Perkembangan ini memang sangat menguntungkan dan memudahkan pihak pemilik *website* untuk mempromosikan layanan atau fitur yang ditawarkan pada *client*. Begitu juga dari sisi *client*, kemudahan akses dengan hanya bermodalkan *pc* dan akses internet dapat menghilangkan keterbatasan akan ruang dan waktu.

Seiring kemajuan teknologi internet, ada sebuah kebutuhan yang mengikuti hal tersebut yaitu kebutuhan akan keamanan data pada *computer* yang sedang melakukan komunikasi melalui internet. Sebuah *computer* menjadi transparan atau mudah diakses dan beresiko untuk mengalami penyusupan dari pihak-pihak yang menginginkan untuk mengakses sistem komputer. Akibat ketransparan tersebut, sistem komputer secara langsung beresiko terhadap ancaman dan serangan. Hal ini sangat berbahaya bagi sistem komputer yang dimiliki sebuah korporasi atau perusahaan yang memiliki data-data yang sangat rahasia dan hanya boleh diakses oleh orang tertentu saja. Dapat kita bayangkan, dalam sekejap data tersebut bisa diakses dan dimanipulasi oleh pihak yang tidak memiliki hak akses. Bentuk lain dari ancaman yang sering terjadi yaitu pencurian terhadap data yang berharga dan rahasia, serta penyadapan terhadap data yang dikirim. Jika ruang lingkup *system* komputer hanya bersifat lokal, ancaman keamanan jaringan bisa terjadi dari ‘orang dalam’ yang berkeinginan untuk melakukan penyusupan dan melakukan ancaman terhadap sistem komputer. Dari hal-hal diatas, maka, dibutuhkan sebuah alat yang dikenal dengan nama *Intrusion Detection Systems (IDS)* atau *system* pendeteksi penyusupan.

Perkembangan metode dan *tool* penyusupan jaringan juga sangat mempengaruhi kebutuhan akan adanya sebuah *IDS* atau sistem pendeteksian penyusupan. Sehingga *IDS* dapat memberikan antisipasi dini terhadap resiko penyusupan. Ada berbagai jenis metode penyusupan misalnya *Denial of Service (DoS), Ping of Death, UDP Bomb* dan *tools* lainnya. Metode tersebut memiliki tujuan dalam menganggu proses kerja suatu *resource* atau jaringan sehingga tidak dapat bekerja sebagaimana mestinya.

Sebuah *Intrusion Detection System (IDS*) merupakan perangkat yang memiliki kemampuan untuk melakukan pendeteksian terhadap serangan dan ancaman yang terjadi pada sebuah jaringan komputer baik yang terhubung dengan internet maupun hanya jaringan lokal. *IDS* memiliki peran yang sangat besar dalam mengamankan kerahasian suatu data dalam sebuah jaringan komputer.

Ada beberapa jenis *IDS* berdasarkan tempat dilakukannya implementasi *IDS*, yaitu *Host Intrusion Detection (HIDS) dan Network Intrusion Detection System (NIDS)*. Kedua jenis *IDS* diatas juga memiliki kelebihan dan kekurangan yang apabila diintegrasikan maka akan memperkecil kekurangan dan menguatkan kelebihan yang dimiliki masing-masing tipe. Salah satu kelebihan *NIDS* yaitu dapat mendeteksi serangan yang tidak terdeteksi oleh *HIDS*. Begitu juga sebaliknya dengan *HIDS*. Kekurangan dari *NIDS* yaitu keterbatasan dalam me-*monitoring host* secara lebih detail, sedangkan *HIDS* tidak mampu melakukan *monitoring traffic* secara keseluruhan.

Dalam melakukan pendeteksian sebuah *IDS* menggunakan dua metode *Signatures* dan *Anomaly Detection. Signatures* merupakan pembandingan antara *rules* atau aturan antara sebuah *traffic* yang sedang dideteksi dengan *traffic* yang mengidentifikasikan terjadinya penyerangan. Sedangkan *Anomaly* ialah perbandingan antara *rules* yang berisi *traffic* normal dengan *traffic* yang sedang dideteksi. *Signature*s memiliki kelemahan dalam menidentifikasi serangan yang tidak didefinisikan pada *rules* yang berisi informasi ciri-ciri penyerangan, sehingga IDS yang menggunakan metode *signatures* tidak mengenali penyerangan baru yang belum terdapat pada *rules*. Jika ditinjau pada metode

Anomaly, maka pendeteksi pun bisa dimanipulasi agar bisa mendeteksi serangan yang belum terdapat pada *rules Signatures*.

Telah banyak *Intrusion Detection System* yang berkembang disaat ini, yaitu *RealSecure* dari *Internet Security Systems (ISS)*, *Cisco Secure Intrusion Detection System* dari *Cisco Systems* (yang mengakuisisi *WheelGroup* yang memiliki produk *NetRanger*), *eTrust Intrusion Detection* dari *Computer Associates* (yang mengakusisi MEMCO yang memiliki *SessionWall-3*), *Symantec Client Security* dari *Symantec*, *Computer Misuse Detection System* dari *ODS Networks*, *Kane Security Monitor* dari *Security Dynamics*, *Cybersafe*, *Network Associates*, *Network Flight Recorder*, *Intellitactics*, *SecureWorks*, *Snort*, *Security Wizards*, *Enterasys Networks*, *Intrusion.com*, *IntruVert*, *ISS*, *Lancope*, *NFR*, *OneSecure*, *Recourse Technologies*, dan *Vsecure.*

Namun ada sebuah *IDS* yang berbasiskan *open source* yaitu *Snort*. *Snort* hanya memanfaatkan konsep pendeteksian yang berdasarkan *Network-based* yang hanya melakukan penganalisaan terhadap paket data yang dianggap sebagai serangan yang melintasi *network*. Sedangkan pendeteksian secara *host*-based yang dikenal dengan nama *Host Intrusion Detection System (HIDS)* merupakan hal yang penting dilakukan agar keamanan *host* tempat *resources* memiliki faktor keamanan. Berdasarkan jenis metode analisa *rules* yang dikategorikan sebagai penyusupan, *Snort* menggunakan metode analisa *signatures* dan *anomaly detection.*

Berdasarkan latar belakang diatas, maka dirancanglah sebuah *IDS* yang berbasiskan *Hybrid Detection System, Signatures dan Anomaly Detection*.

1.2 Rumusan Masalah

Permasalahan yang dirumuskan berdasarkan latar belakang yang telah disampaikan ialah bagaimana cara merancang dan mengimplementasikan konsep *Hybrid Detection System, Sigantures and Anomaly Detection* untuk sebuah Sistem pendeteksi penyusupan dengan *active response*.

1.3 Batasan Masalah

Batasan masalah dari penyusunan tugas akhir ini adalah :

1. Protokol yang digunakan yaitu *TCP*, *UDP*, dan *ICMP*
2. Fitur yang dimiliki oleh sistem ini difokuskan kepada tiga bagian, sebagai berikut :
    a. *Sniffer mode*, untuk melihat paket lewat di *network*
    b. Paket *Logger mode*, untuk mencatat semua paket yang lewat di *network* dan akan dilakukan penganalisaan di masa mendatang
    c. *HyDManSys Mode* yang terdiri atas dua sub bagian:
        i. *Network Intrusion detection mode*, berfungsi untuk mendeteksi serangan yang melalui jaringan
        ii. *Host Intrusion detection mode*, berfungsi untuk mendeteksi serangan yang terjadi pada suatu *host*
3. Indikator yang digunakan dalam analisa penyusupan jaringan ialah *Internet Protokol Address (IP Address)* dan *port number*

1.4 Tujuan

Tujuan dari pembuatan Sistem Pendeteksi Penyusupan ini adalah Membangun sebuah *IDS* yang mampu mengamankan secara aktif, *traffic* jaringan dan *host* dari resiko penyusupan dengan mengintegrasikan konsep *Hybrid Detection System , Signatures* dan *Anomaly detection* secara optimal.

1.5 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir terdiri dari lima bagian. Penjelasan mengenai kelima bagian ini, yaitu :

BAB I PENDAHULUAN

Bab ini menjelaskan dasar-dasar dari penulisan laporan tugas akhir ini, yang terdiri dari latar belakang, rumusan masalah, batasan masalah, tujuan, serta sistematika penulisan laporan tugas akhir.

BAB II LANDASAN TEORI

Bab ini membahas teori-teori yang berhubungan dengan spesifikasi pembahasan penelitian yang akan diangkat, yang terdiri dari pembahasan mengenai Penyusupan, Sistem Pendeteksi Penyusupan, Protokol-protokol yang digunakan oleh sistem, *Snort* dan *Java Security*

BAB III METODOLOGI PENELITIAN

Bab ini membahas langkah-langkah yang dilaksanakan dalam proses penelitian, yaitu metode pengembangan sistem, tahapan penelitian, pengumpulan data, analisa sistem, perancangan sistem dan implementasi beserta pengujian pada Sistem Pendeteksian Penyusupan.

BAB VI ANALISA DAN PERANCANGAN

Bab ini membahas analisis sistem, hasil analisa, deskripsi sistem, fungsi produk, karakteristik pengguna, deskripsi umum kebutuhan, perancangan *database*, perancangan antar muka dan struktur menu pada Sistem Pendeteksian Penyusupan.

BAB IV IMPLEMENTASI DAN PENGUJIAN

Bab ini membahas implementasi Sistem Pendeteksian Penyusupan dan pengujiannya.

BAB V PENUTUP

Bab ini berisi kesimpulan yang dihasikan dari pembahasan tentang Sistem Pendeteksian Penyusupan yang telah dilaksanakan dan beberapa saran sebagai hasil akhir dari penelitian yang telah dilakukan.

BAB II
LANDASAN TEORI

2.1 *Penyusupan Jaringan*

Penyusupan Jaringan merupakan suatu pengaksesan yang dilakukan seseorang atau komputer yang tidak memiliki hak otorisasi terhadap sumber daya yang akan diakses. Ada berbagai tipe penyusupan yang terjadi pada jaringan, sebagai berikut (Berbagai,2008):

1. *LAND Attack*

Adalah salah satu macam serangan terhadap suatu server yang terhubung dalam suatu jaringan untuk menghentikan layanan, sehingga terjadi gangguan terhadap layanan atau jaringan komputer . Tipe serangan semacam ini disebut sebagai *Denial of Service (DoS) attack*. *LAND attack* dikategorikan sebagai serangan *SYN (SYN attack)* karena menggunakan packet *SYN (synchronization)* pada waktu melakukan *3-way handshake* untuk membentuk suatu hubungan berbasis *TCP/IP*.

2. *Ping of Death*

Ping of Death merupakan suatu serangan (*Denial of Service) DoS* yang memanfaatkan fitur yang ada di *TCP/IP* yaitu *packet fragmentation* atau pemecahan paket. Penyerang dapat mengirimkan berbagai paket *ICMP* (digunakan untuk melakukan ping) yang terfragmentasi sehingga waktu paket-paket tersebut disatukan kembali, maka ukuran paket seluruhnya melebihi batas 65536 *byte*.

3. *Teardrop*

Teardrop attack adalah suatu serangan bertipe *Denial of Service (DoS)* terhadap suatu *server* atau komputer yang memanfaatkan fitur yang ada di *TCP/IP* yaitu *packet fragmentation* atau pemecahan paket, dan kelemahan yang ada di *TCP/IP* pada waktu paket-paket yang terfragmentasi tersebut disatukan kembali.

Dalam suatu pengiriman data dari satu komputer ke komputer yang lain melalui jaringan berbasis *TCP/IP*, maka data tersebut akan dipecah-pecah menjadi beberapa paket yang lebih kecil di komputer asal, dan paket-paket tersebut dikirim dan kemudian disatukan kembali di komputer tujuan. Server bisa diproteksi dari tipe serangan *teardrop* ini dengan *paket filtering* melalui *firewall* yang sudah dikonfigurasi untuk memantau dan memblokir paket-paket yang berbahaya.

4. *Half-Open Connection*

Dalam serangan *half-open connection*, penyerang mengirimkan ke server yang hendak diserang banyak paket *SYN* yang telah di-*spoof* atau direkayasa sehingga alamat asal (*source address)* menjadi tidak valid. . Tipe serangan *half-open connection* atau *SYN* attack ini dapat dicegah dengan paket filtering dan *firewall*, sehingga paket-paket *SYN* yang invalid tersebut dapat diblokir oleh *firewall* sebelum membanjiri *server*.

5. *UDP Bomb Attack*

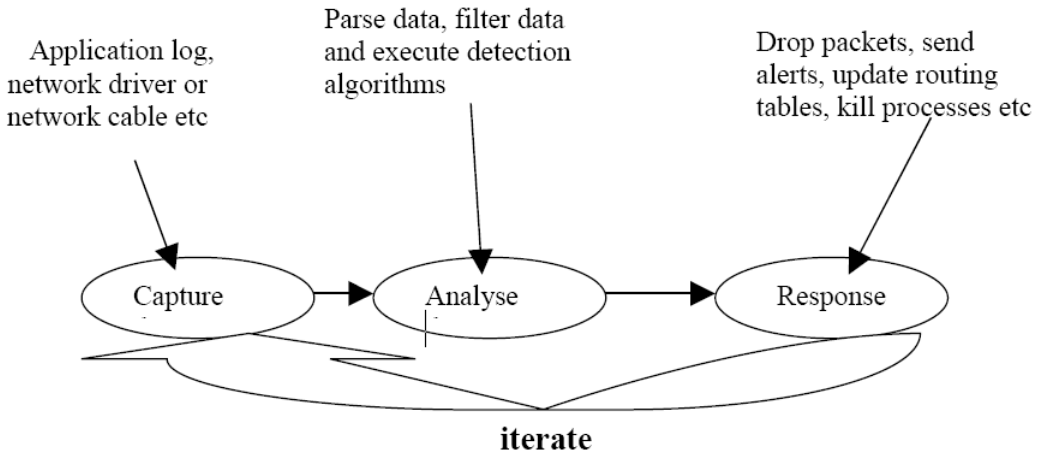
Untuk melakukan serangan *UDP Bomb* terhadap suatu *server*, seorang penyerang mengirim sebuah paket *UDP (User Datagram Protocol)* yang telah di-*spoof* atau direkayasa sehingga berisikan nilai-nilai yang tidak *valid* di *field* tertentu. Jika *server* yang tidak terproteksi masih menggunakan sistem operasi (*operating system*) lama yang tidak dapat menangani paket-paket *UDP* yang tidak valid ini, maka *server* akan langsung *crash*.

2.2 *Definisi Sistem Pendeteksian Penyusupan*

Sistem Pendeteksian Penyusupan, lebih dikenal dengan nama *IDS (Intrusion Detection System)*. Sistem ini merupakan usaha mengidentifikasi adanya penyusup yang memasuki sistem tanpa otorisasi (misal *cracker*) atau seorang user yang sah tetapi menyalahgunakan *privilege* sumberdaya sistem. *IDS*

akan melakukan pemberitahuan saat mendeteksi sesuatu yang dianggap sebagai mencurigakan atau tindakan ilegal.

Konsep dasar dari *IDS* ialah melakukan pengumpulan data dan identifikasi apakah suatu akses yang dilakukan terhadap jaringan atau *host* (*resources*), dapat dikategorikan sebagai serangan yang telah didefinisikan pada rules atau aturan *IDS*. Selanjutnya *IDS* akan memberikan peringatan kepada *user* bahwa telah terjadi penyerangan pada jaringan dan *host* yang dilindungi. Konsep *IDS* dapat terlihat pada gambar dibawah ini :



Gambar 2.1 Konsep *IDS*

Berdasarkan cara kerja penganalisaan paket data yang dilakukan *IDS*, ada dua tipe dasar *IDS*, yaitu

1) *Knowledge-based atau misuse detection (signature)*

Knowledge-based IDS dapat mengenali adanya penyusupan dengan cara menyadap paket data kemudian membandingkannya dengan *database rule IDS* (berisi *signature - signature* paket serangan). Jika paket data mempunyai pola yang sama dengan (setidaknya) salah satu pola di *database rule IDS*, maka paket tersebut dianggap sebagai serangan, dan demikian juga

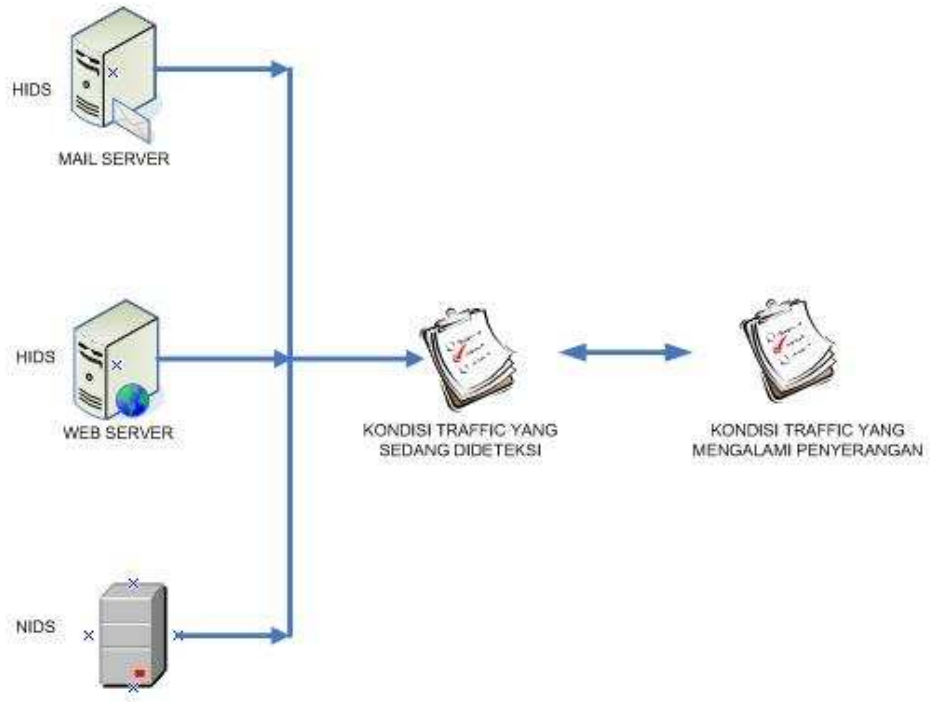
sebaliknya, jika paket data tersebut sama sekali tidak mempunyai pola yang sama dengan pola di *database rule IDS*, maka paket data tersebut dianggap bukan serangan.

Konsep *Signatures* : Membandingkan antara pola yang telah didefinisikan pada sistem pendeteksi penyusupan dengan kondisi *traffic* yang sedang dideteksi. Apabila terjadi kesamaan antara kedua hal tersebut, maka diindikasikan terjadinya penyerangan.

***Rules Signatures* :** berisi informasi mengenai jenis-jenis serangan, misalnya contoh *rules* Snort yang mengindentikasikan adanya penyerangan pada protokol *HTTP* , sebagai berikut :

```
alert tcp $EXTERNAL NET
alert tcp $EXTERNAL NET any -> $HTTP SERVERS $HTTP
PORTS
(msg:"WEB-IIS Unicode directory traversal attempt";
flow:to server, established; content:¨/..%c0%af../";
nocase; classtype:web-application-attack;
reference:cve,
CVE-2000-0884; sid:981; rev:6;)
```

Rules di atas terdiri dari 2 bagian: *header* dan *option*. Bagian *”alert tcp $EXTERNAL NET any -¿ $HTTP SERVERS $HTTP PORTS”* adalah *header* dan selebihnya merupakan *option*. Dari *rules* seperti di ataslah, *IDS* mengidentikasi apakah sebuah paket data dianggap sebagai penyusupan atau serangan. Paket data dibandingkan dengan *rules IDS*, apabila terdapat dalam *rule*, maka paket data tersebut dianggap sebagai penyusupan atau serangan dan demikian juga sebaliknya. Jika tidak ada dalam *rules* maka dianggap bukan penyusupan atau serangan.



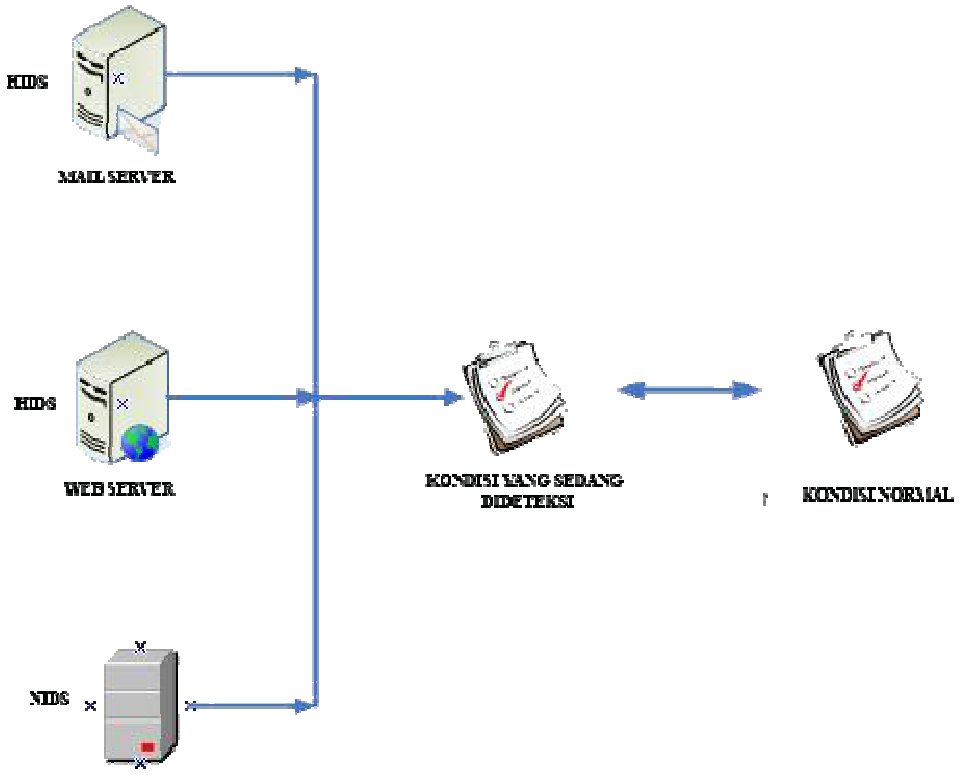
Gambar 2.2 *Signatures Detection*

2) *Behavior based (anomaly)*

IDS jenis ini dapat mendeteksi adanya penyusupan dengan mengamati adanya kejanggalan-kejanggalan pada sistem, atau adanya penyimpangan-penyimpangan dari kondisi normal, sebagai contoh ada penggunaan memori yang melonjak secara terus menerus atau ada koneksi paralel dari 1 buah IP dalam jumlah banyak dan dalam waktu yang bersamaan. Kondisi-kondisi diatas dianggap kejanggalan yang kemudian oleh *IDS* jenis *anomaly based* dianggap sebagai serangan.

Konsep *Anomaly* : Melakukan perbandingan antara kondisi normal *traffic* dengan kondisi yang sedang dideteksi. Apabila terjadi kesamaan antara kedua hal diatas maka *traffic* dinyatakan dalam keadaan normal. Namun apabila sebaliknya, maka *traffic* dikategorikan sebagai *traffic* yang sedang mengalami penyusupan atau penyerangan.

Rules Anomaly : Berupa informasi tentang identifikasi *Data flow*, definisi *Allowed Category* (kategori paket yang diizinkan), definisi *Suspicious Category* (katagori paket yang dicurigai), definisi *Not Allowed Category* (kategori paket yang tidak diizinkan). Keempat bagian tersebut dijadikan acuan dalam mengidentifikasi sebuah serangan. Data flow berisi informasi mengenai *Source* dan *Destination IP* atau *Network, Network protocol, Application Protocol, Protocol options, content (size, type, characteristic string), TCP flags, TTL* dan sebagainya.



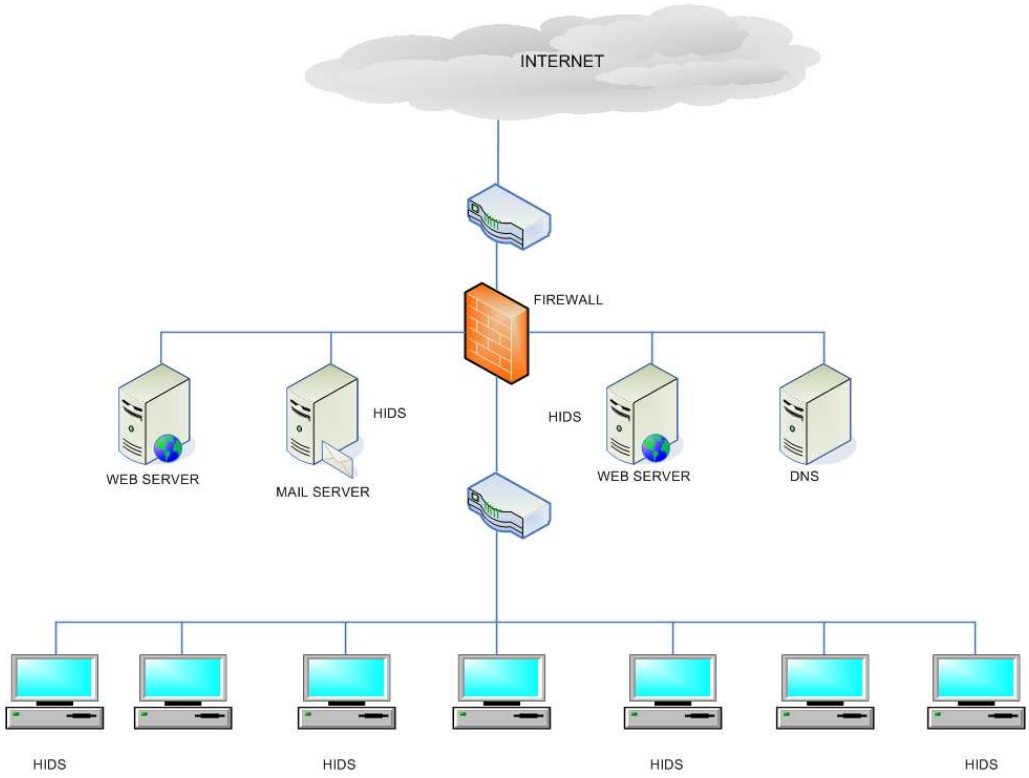
Gambar 2.3 ***Anomaly Detection***

Jika dilihat dari kemampuan mendeteksi penyusupan jaringan, ada tiga jenis *IDS* yang akan digunakan, yaitu :

1) Host based intrusion detection system (HIDS)

Host based mampu mendeteksi hanya pada host tempat implementasi *IDS*. Konsep *Host based* hanya mendeteksi suatu *host resource* pada *log* yang dihasilkan dengan memonitor sistem *file, event* dan keamanan pada Windows

NT dan *syslog* pada sistem operasi UNIX. Apabila *log* tersebut merupakan suatu penyerangan maka akan dilaksanakan peringatan pada *IDS* (Ariyus,2007).

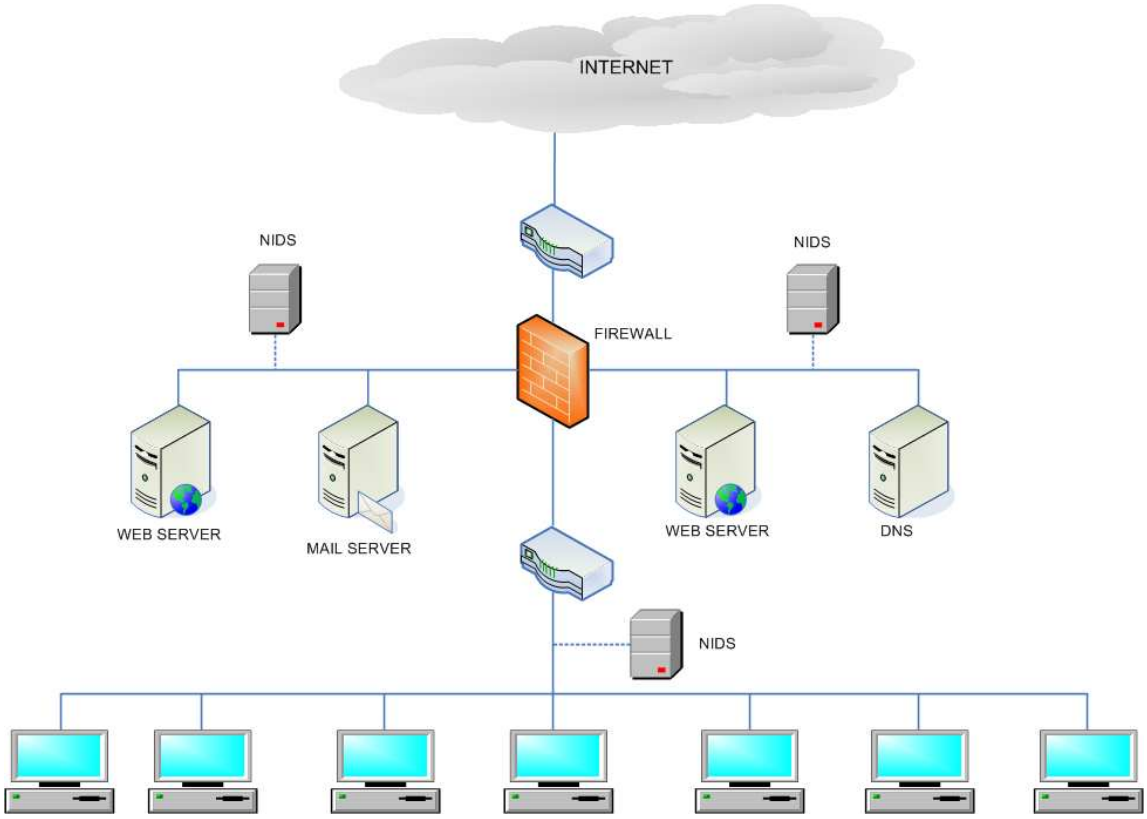


Gambar 2.4 *HIDS*

2) Network based intrusion detection system (NIDS)

Network based IDS mampu mendeteksi seluruh host yang berada satu jaringan dengan *host* implementasi *IDS* tersebut.

Konsep NIDS : mengumpulkan paket data yang akan dipantau oleh Network Adapter, kemudian memonitor dan menganalisa paket data yang ada pada jaringan (Ariyus, 2007).



Gambar 2.5 NIDS

3) *Hybrid Intrusion Detection System*

Hybrid Intrusion Detection System merupakan gabungan antara kemampuan *NIDS* dan *HIDS*. Perpaduan kedua *IDS* menghasilkan sebuah *Hybrid IDS* yang mampu melakukan pendeteksian pada *resource host* suatu jaringan dan *traffic* jaringan secara menyeluruh.

Berikut merupakan kelebihan yang dimiliki *HIDS* dan *NIDS*, yaitu :

Tabel 2.1 Kelebihan *HIDS* dan *NIDS*

Tipe *Intrusion Detection System*	**Kelebihan**
Host Intrusion Detection System	Dapat menguji akurasi keberhasilan atau kegagalan serangan
	Dapat memonitor aktivitas tertentu
	Dapat mendeteksi serangan yang lolos dari *NIDS*
	Cocok untuk lingkungan yang menggunakan *encrypt* dan *switch*
	Deteksi dan respon secara *near-real time*
	Tidak memerlukan tambahan *hardware*
Network Intrusion Detection System	Biaya yang lebih rendah
	Dapat mendeteksi serangan yang tidak terdeteksi oleh *HIDS*
	Kesulitan bagi penyerangan untuk menghapus jejak
	Dapat mendeteksi dan merespon secara *real time*
	Deteksi serangan yang gagal dan kecendrungan serangan
	Tidak bergantung pada sistem operasi

HIDS dan *NIDS* juga memiliki kelemahan, sebagai berikut :

Tabel 2.2 Kelemahan *HIDS* dan *NIDS*

Tipe *Intrusion Detection System*	**Kelemahan**
Host Intrusion Detection System	Manajemen yang rumit, informasi harus dikonfigurasi untuk setiap *host* yang ada
	Sedikit sumber informasi dan apabila *HIDS* dilumpuhkan penyerang, maka akses host dapat diambil alih penyerang
	Tidak cocok untuk *monitoring* jaringan
	Dapat dilumpuhkan dengan serangan *Denial of Service*
	Menggunakan sistem operasi audit sebagai sumber informasi, sehingga menambah *space harddisk*

Tipe *Intrusion Detection System*	Kelemahan
Host Intrusion Detection System	Menggunakan sumber daya komputer, sehingga menambah pemakaian sumber daya komputer *host* yang dideteksi
Network Intrusion Detection System	Gagal mendeteksi jaringan yang sangat sibuk karena keterbatasan pemrosesan paket yang besar
	Banyak keuntungan *NIDS* tidak berlaku untuk jaringan dengan *metode switch-based* yang lebih modern
	Tidak memonitor paket yang dienkripsi
	Tidak bisa memberitahukan kondisi serangan yang berhasil mendapatkan sistem dan terbatas pada pemberitahuan bahwa sedang terjadi serangan
	Bermasalah dengan serangan yang menggunakan paket fragmentasi

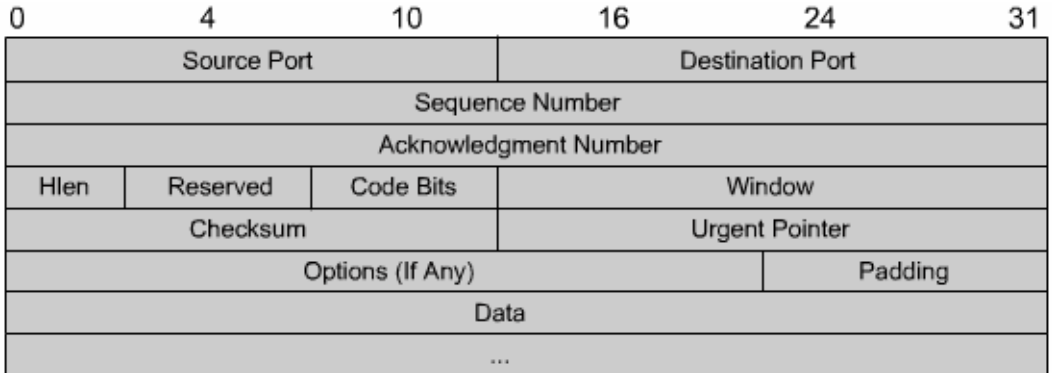
2.3 *Protokol-protokol yang digunakan pada penelitian*

Protokol merupakan media antar perangkat *(devices)* dalam berkomunikasi. Apabila akan dilakukan pertukaran informasi antar beberapa komputer dalam suatu jaringan, maka protokol-lah yang menjadi media komunikasi. Ada berbagai jenis protokol, namun pada penelitian ini akan difokuskan kepada protokol *TCP*, *UDP*, dan *ICMP*. Uraian mengenai ketiga jenis protokol diatas, yaitu :

2.3.1 TCP (Transmission Control Protocol)

TCP merupakan *protokol connection-oriented*. Komunikasi data antar *host* terjadi melalui proses sinkronisasi untuk membentuk *virtual connection* setiap *session* antar *host*. Proses sinkronisasi ini meyakinkan kedua sisi apakah sudah siap transmisi data atau belum dan mengizinkan *device* untuk menentukan inisial *sequence number*. Proses ini disebut dengan *3-way handshake*. Untuk membentuk

koneksi *TCP*, klien harus menggunakan nomor port tertentu dari layanan yang ada di server (Jaringan,-)

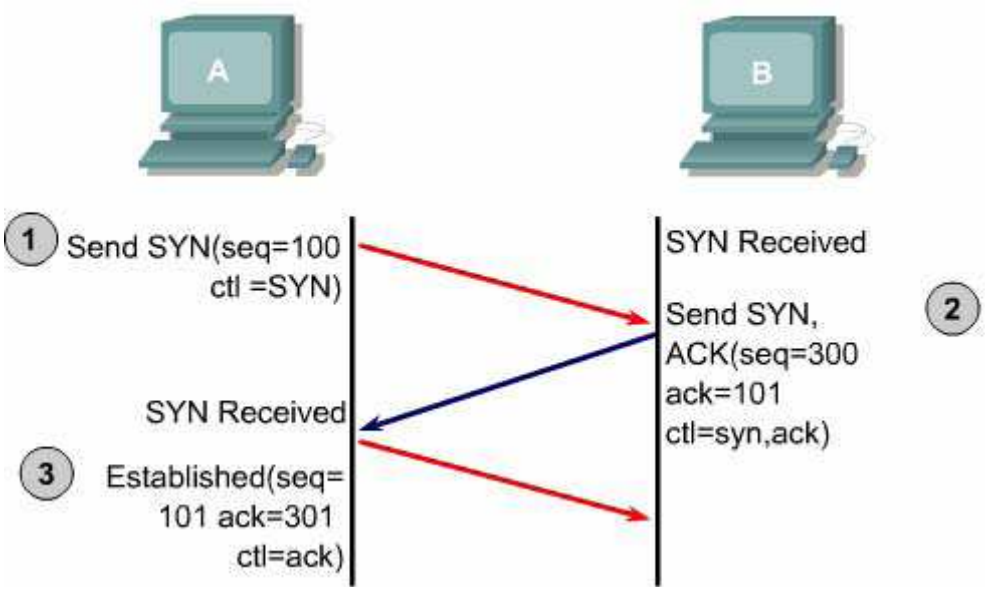


Gambar 2.6 : Format TCP

Tahap pertama, klien mengirimkan paket sinkronisasi (*SYN flag* set) untuk inisialisasi koneksi. Paket dianggap *valid* kalau nilai *sequence number*-nya misalnya x. bit *SYN* menunjukkan permintaan koneksi. *Bit SYN* panjangnya satu *bit* dari segmen *header* TCP. Dan *sequence number* panjangnya 32 *bit* (Jaringan,-).

Tahap kedua, *host* yang lain menerima paket dan mencatat sequence number x dari klien dan membalas dengan acknowledgement (*ACK flag set*). *Bit control ACK* menunjukkan bahwa *acknowledgement number* berisi nilai *acknowledgement* yang valid. *ACK flag* panjangnya satu bit dan Ack number 32 bit dalam segmen *TCP header*. Sekali koneksi terbentuk, *ACK flag* diset untuk semua segmen. (Jaringan,-).

Tahap ketiga, klien meresponnya dengan *Ack Number* y + 1 yang berarti ia menerima *ack* sebelumnya dan mengakhiri proses koneksi untuk *session* ini(Jaringan,-).



Gambar 2.7 : Proses *3-way Handshake TCP*

2.3.2 *UDP (Unit Datagram Protocol)*

UDP adalah protokol *connectionless*, transmisi paket data tidak dijamin sampai ke tujuan. Berikut beberapa hal menyangkut *UDP* :

a) Tidak perlu adanya setup koneksi terlebih dahulu (hal ini dapat menyebabkan tambahan *delay*)
b) sederhana, artinya antara penerima dan pengirim tidak perlu menjaga session atau status koneksi
c) ukuran *header* segment sederhana
d) tidak perlu kontrol kemacetan koneksi, artinya *UDP* dapat mengirimkan per segment tanpa dipengaruhi oleh kesibukan jaringan

# of Bits	16	16	16	16	16
	Source Port	Destination Port	Length	Check Sum	Data...

Gambar 2.8 : Segmen UDP

2.3.3 ICMP *(Internet Control Message Protocol)*

Internet Control Message Protocol (ICMP) berfungsi untuk memberitahu pengirim bahwa terjadi *error* data pada proses pengiriman.

Pesan *ICMP* ditransmisikan menggunakan protokol *IP* dengan metode pengiriman *unreliable*. Tidak ada proses untuk menentukan masalah saat pengiriman data ke jaringan. Jika terdapat kegagalan seperti *router* mati, atau jika *device* tujuan tidak terhubung ke jaringan, maka data tidak dapat terkirim. *ICMP* merupakan komponen dari protokol *TCP/IP* yang membantu *IP* untuk mengidentifikasi kesalahan-kesalahan itu.

Contoh penggunaan *ICMP* yaitu (Ariyus, 2007):

a. Apabila datagram tidak mampu mencapai tujuan dalam mentransfer paket
b. Apabila *router* tidak memiliki kapasitas penyangga untuk menjalankan *datagram*
c. Apabila *router* tidak bisa mengalihkan *host* untuk mengirim paket pada jalur yang pendek

Destination Unreacheble merupakan salah satu contoh pesan *ICMP*. Hal ini terjadi ketika *router* tidak mengetahui tujuan permintaan pengiriman paket yang di-*request* oleh *host*, sebagai balasan *router* memberikan pesan *Destination Unreacheble*.

2.4 *Snort*

Snort merupakan sebuah *Network Intrusion Detection system* yang berbasiskan *Open source*. Snort pertama kali dikembangkan oleh *Martin Roesch*, 1998. Situs www.snort.org, menyediakan *snort* secara gratis. *Snort* mendukung berbagai sistem operasi seperti Windows, Linux, BSD, Solaris dan lainnya.

Snort dioperasikan pada tiga jenis mode, sebagai berikut (Ariyus, 2007) :

a. Paket *Sniffer*, berfungsi untuk melihat paket yang melewati jaringan

b. Paket *Logger Mode* berfungsi untuk mencatat semua paket yang melewati jaringan dan dianalisa di masa mendatang
c. *NIDS*, berfungsi untuk mendeteksi serangan yang terjadi pada suatu jaringan

Snort memiliki beberapa komponen dasar, anatara lain (Ariyus,2007) :

a. *Decoder*, berfungsi untuk menyesuaikan paket yang di-*capture* dalam bentuk struktur data dan mengidentifikasi protokol, *decode IP*, *TCP*, *UDP* tergantung informasi yang dibutuhkan seperti *port number* dan *IP address*. Apabila ditemukan paket yang cacat, maka *snort* akan memberikan peringatan.
b. *Preprocessors*, berfungsi sebagai saringan yang mengidentifikasikan berbagai hal yang akan dipperiksa oleh *Detector engine* serta berfungsi dasar untuk mengambil paket yang berpotensi berbahaya, agar dapat dikenali polanya oleh *Detector Engine.*
c. *Global Section*, berfungsi untuk mengizinkan *mapping* file IIS *Unicode*, konfigurasi peringatan untuk *proxy server* dengan *proxy alert*, konfigurasi *HTTP* pada *nonauthorized port* dengan menggunakan *stect_anomoulus_traffic.*
d. *Server Section*, berfungsi untuk mengizinkan *setting HTTP server profiles* yang berbeda, konfigurasi tipe serangan dan nomalisasi berdasarkan *server* yang ada.
e. *Rules*, ialah suatu file teks yang berisi daftar aturan sintaks-nya telah diketahui. Rules meliputi *protocol address*, *output plug-ins*, dan hal yang berhubungan dengan penyusupan atau serangan.
f. *Detection Engine*, berfungsi dalam menemukan dan menginisialisasi sebuah paket merupakan serangan
g. *Output plugs-in*, berfungsi sebagai modul yang mengatur format keluaran untuk *alert* dan *file logs* yang bisa diakses dengan menggunakan *console, extern files, database* dan lainnya

Gambar 2.10 : Proses Snort

2.5 *Java*

Java adalah sebuah bahasa pemrograman yang dikembangkan sebagai komponen utama *platform Java* oleh *Sun Microsystems* dan diluncurkan pada tahun 1995. Bahasa pemrograman ini banyak dipengaruhi oleh bahasa pemrograman sebelumnya yakni C dan C++. Beberapa paket yang disertakan dalam peluncuran awalnya adalah sbb: Java.lang ,Java.io, Java.util, Java.net, Java.awt, Java.applet. (Java,2008)

2.5.1 Java Runtime Environment

Java Runtime Environment, (JRE) merupakan sebuah perangkat lunak yang dibutuhkan untuk menjalankan semua aplikasi yang berbasis *Java Platform*. JRE banyak sekali digunakan sebagai *plug-ins web browser* dan dalam berbagai program kontemporer. *Sun Microsystem* juga meluncurkan *superset* dari *JRE* dan diberi nama *Java 2 SDK*, yang sering disebut *JDK*. Dalam *JDK* ini terdapat beberapa komponen pengembangan *Java*, seperti : *Java Compiler*, *Javadoc*, *Jar* dan *debugger*. Salah satu kelebihan yang ditawarkan oleh JRE adalah kesalahan-kesalahan (*exceptions*) yang terjadi tidak akan membuat sistem menjadi *crash* atau *hang*. Kelebihan lainnya adalah terdapatnya komponen yang mampu merekam secara tepat waktu segala kesalahan yang terjadi ke dalam memori. Komponen ini dinamakan *Automated Exception Handling.* Beberapa komponen lainnya antara lain : (Java,2008)

a. *Library Java* mengkompilasi kode *byte* dari *source code* yang dibentuk oleh *Implementator JRE* untuk mendukung pengembangan aplikasi dalam *Java*, beberapa contoh *Library* dalam *Java* :

   1. The Core Library, yang berisikan :

   - *Library* koleksi yang mengimplementasikan struktur data seperti *List, Dictionaries, Trees* dan *Sets*
   - *Library* Proses XML (*Parsing, Transforming, Validating*)
   - Sekuritas
   - *Library* internasionalisasi dan lokalisasi

2. Library terintegrasi.

3. Library User Interface yang mencakup :

- *Abstract Windowing Toolkit (AWT)*, yang menyediakan komponen GUI untuk meng-"gambar"-kan komponen tersebut dan membuat komponen tersebut mampu menangani *event handling.*
- *Library Swing.*
- *API* untuk *capture audio*, pemrosesan dan *playback.*

b. Implementasi ketergantungan *platform Java Virtual Machine* (*JVM*).

c. *Plug-ins* yang menjalankan *applet* untuk dijalankan di *web browser.*

d. *Java Web Start*, yang menjalankan aplikasi *Java* untuk didistribusikan secara efisien ke pengguna internet.

e. Lisensi dan dokumentasi.

2.5.2 *Java Platform, Standard Edition*

Java Platform, biasanya disebut *Java SE*, adalah sebuah *platform* yang biasa digunakan untuk pemrograman *Java*. *Java SE* umumnya digunakan sebagai *platform* yang bekerja pada aplikasi portable. Dalam prakteknya, *Java SE* tersusun atas mesin virtual yang digunakan untuk mengoperasikan program *Java*, selain itu juga ia tersusun atas serangkaian *Library* (atau paket) yang dibutuhkan untuk pengaksesan *File System*, Jaringan, Antarmuka grafis, dan sebagainya. (*Java*,2008)

2.5.3 *Java Security*

Dalam upaya mendukung pembuatan aplikasi yang memiliki tingkat keamanan tinggi, *Java* menyediakan suatu model *security* yang awalnya dikenal sebagai model *sandbox.* Model ini pada prinsipnya bertugas untuk membatasi aplikasi aplet. Seiring perkembangannya, *Java* memperbaiki model *sandbox* dan menghasilkan arsitektur keamanan yang lebih baik. Fitur-fitur pendukung *security* secara khusus diimplementasikan melalui *API Java Security* dan dicerminkan oleh

paket *java.security*. Paket ini menyediakan koleksi kelas dan *interface* yang mudah untuk dikonfigurasi. (Java,2008)

a. ***Provider***

Kelas ini merepresentasikan *provider* untuk *API Java Security*, dimana *provider* mengimplementasikan beberapa atau semua bagian *Java security*. Layanan – layanan yang diberikan oeh *provider* meliputi algoritma kriptografi, pembentukan key, konversi dan fasilitas pengelolaan

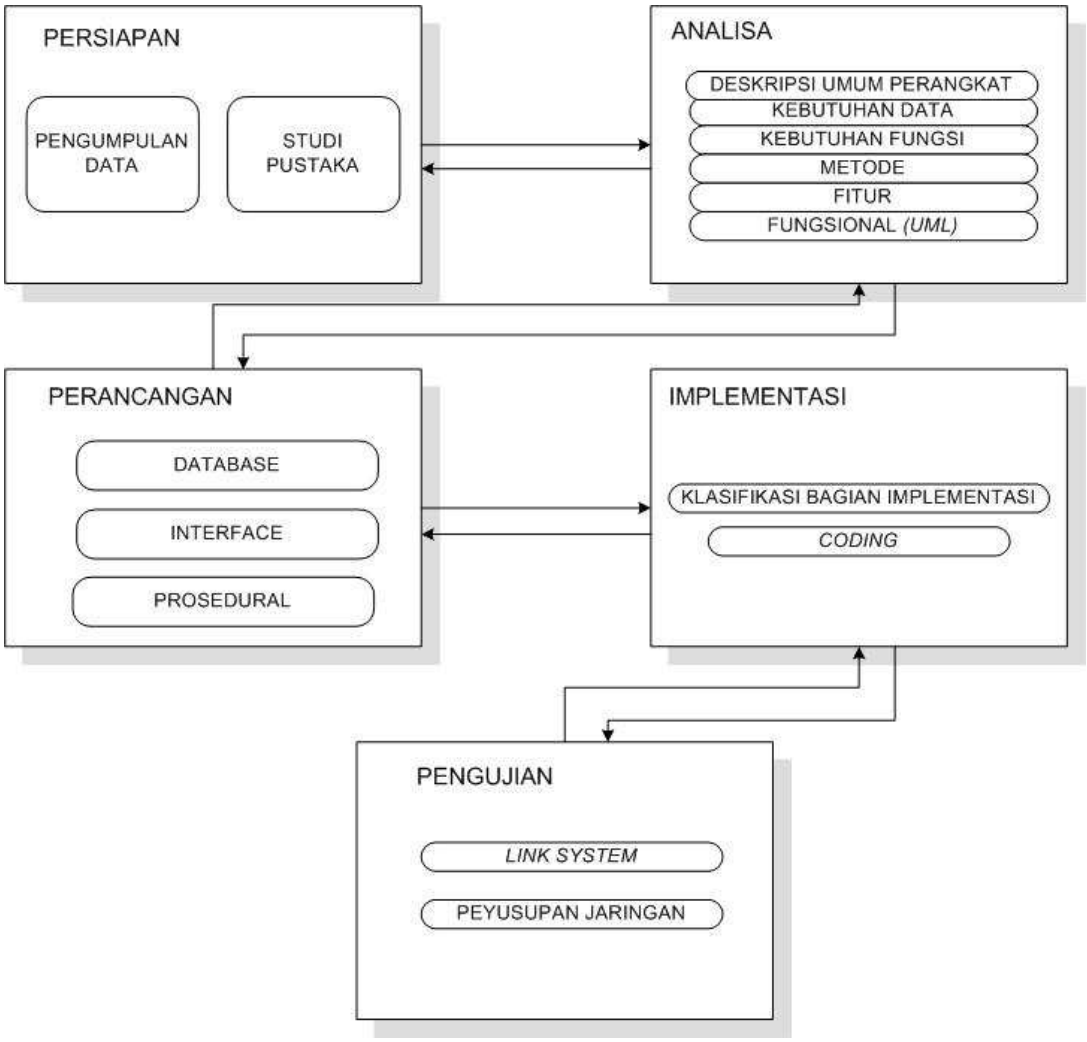
b. ***Message Digest***

Messsage Digest dikenal sebagai kriptografi *checksum* atau *secure hash*. *Message digest* berfungsi untuk meningkatkan keamanan transformasi data, seperti *password*. Dalam implementasinya, nilai *message digest* dibandingkan dengan nilai asli. Paket *java.security* mengimplementasikan *message digest* melalui kelas *Message Digest*. Untuk menghasilkan *message digest*, kita bisa menggunakan algoritma MD5 atau SHA-1.

BAB III
METODOLOGI PENELITIAN

3.1 Tahapan Penelitian

Tahapan Penelitian yang akan dilaksanakan pada pengembangan sistem penyusupan Jaringan Komputer ini dapat terlihat pada gambar dibawah ini :



Gambar 3.1 : Tahapan Penelitian

3.1.1 Pengumpulan Data

Pengumpulan data merupakan tahapan persiapan yang harus dilaksanakan terlebih dahulu sebelum dilakukan penelitian. Berikut merupakan aktivitas yang dilaksanakan dalam pengumpulan data :

1. Studi Pustaka

   Studi pustaka berfungsi untuk mendukung penelitian yang akan dilaksanakan. Pengumpulan teori-teori yang mendukung dalam penelitian ini merupakan kegiatan dalam studi pustaka. Teori-teori bersumber dari buku, jurnal dan penelitian-penelitian sejenis.

2. Wawancara

   Wawancara berfungsi untuk mengumpulkan informasi yang akan berguna untuk tahap analisa dan tahap-tahap selanjutnya. Wawancara dilakukan terhadap praktisi dan akademisi *security* di Teknik Informatika UIN SUSKA RIAU.

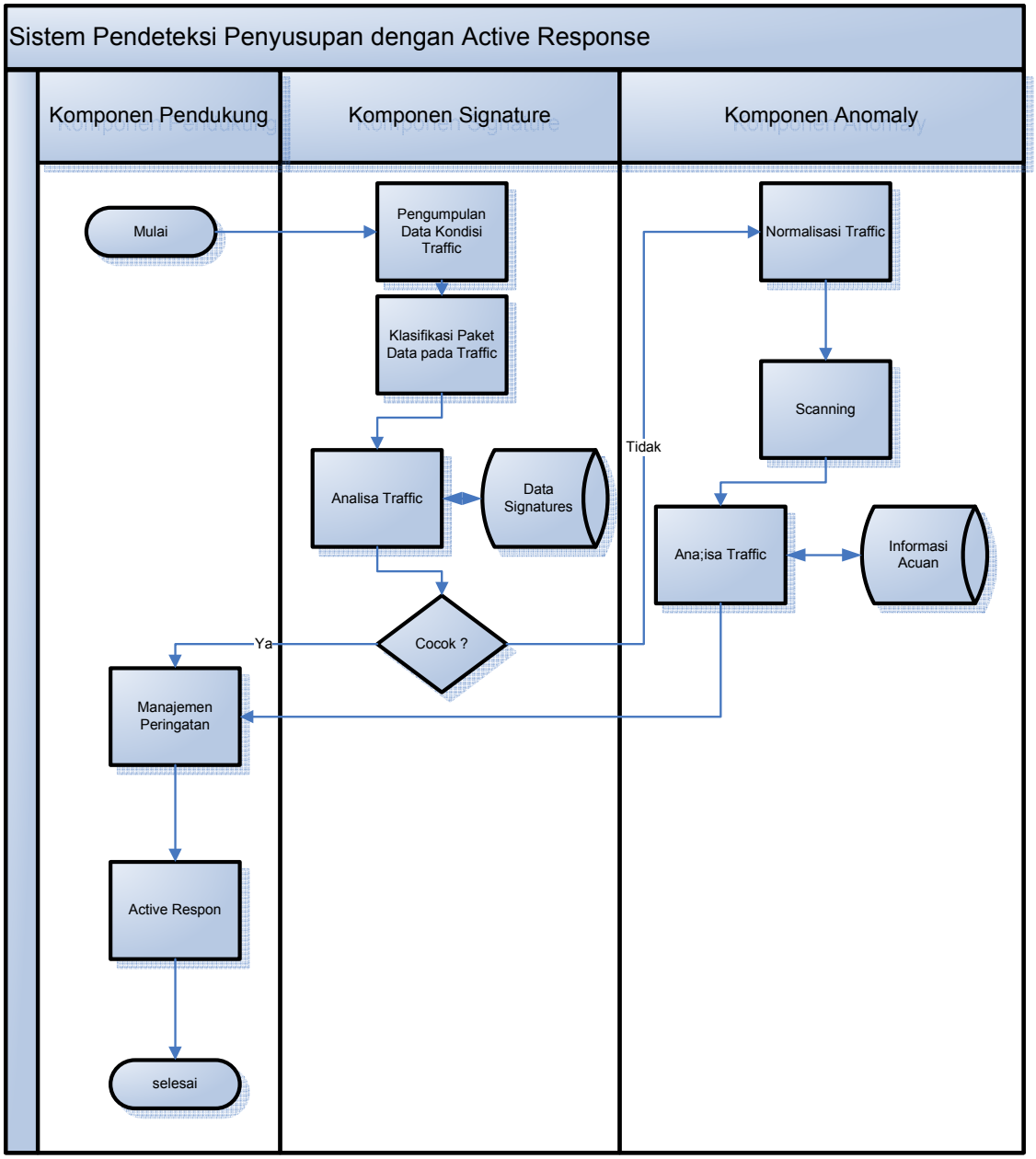
3.1.2 Analisa sistem

Analisa yang dilakukan dalam penelitian ini yaitu sebagai berikut :

1. Deskripsi Umum Perangkat Lunak adalah penggambaran perangkat secara detail agar dapat dipahami setiap fungsi unjuk kerja yang dimiliki perangkat.
2. Analisa Kebutuhan Data, ialah analisa terhadap data yang akan digunakan oleh perangkat.
3. Analisa Kebutuhan Fungsi adalah analisa dalam menentukan fungsi yang dilaksanakan oleh perangkat
4. Analisa Metode merupakan analisa dalam mengintegrasikan metode yang akan digunakan sehingga tercapai tujuan yang diinginkan.

5. Analisa Fitur adalah penentuan fitur yang akan ditampilkan oleh perangkat
6. Analisa Fungsional ialah identifikasi unjuk kerja sistem dari mulai objek, aktor, *usecase, class,* aktivitas yang dilakukan perangkat, runtutan kondisi perangkat, serta pengembangan pernagkat.

Awal dari sistem ini, yaitu penginputan atau pengupdate-an *rules* atau data informasi yang akan digunakan pada proses analisa sebuah kondisi jaringan. Hal ini berkaitan dengan metode yang digunakan yaitu *signature* dan *anomaly*. *Rules* pada *signature* merupakan *rules* yang berisi informasi mengenai ciri-ciri penyusupan, *system log* dan *event* yang mengindikasikan terjadinya serangan atau penyusupan. Sedangkan pada metode *anomaly* merupakan *rules* yang berisi informasi acuan tentang kondisi normal suatu jaringan atau host. Apabila terjadi kecocokan antara *rules* dengan keadaan yang sedang terjadi, maka akan dilakukan peringatan, kemudian respon aktif terhadap penyusupan tersebut. Tujuan Akhir dari pembuatan sistem ini adalah menghasilkan sebuah respon aktif terhadap penyusupan yang terjadi pada jaringan dan *host* yang akan dilindungi . Analisa sistem dapat terlihat pada flow chart dibawah ini :



Gambar 3.2 *Flow Chart* Sistem Pendeteksian Penyusupan Jaringan dengan *Active Response* menggunakan metode *Hybrid Intrusion Detection, Signatures* dan *Anomaly Detection*

3.1.3 Perancangan Sistem

Perancangan system merupakan kegiatan merancang kebutuhan data dengan perancangan *database*, struktur menu dan antar muka (*user interface*).Perancangan *database* meliputi kegiatan membangun *database*, tabel, atribut, *primary key, type data* yang dimiliki setiap atribut, serta relasi antar tabel *database*. Perancangan struktur menu meliputi penempatan menu-menu yang sesuai dengan kategori tampilan yang dimiliki sistem. Perancangan tampilan meliputi pembuatan antar muka *user* dan unjuk kerja perangkat.

3.1.4 Implementasi

Tahap ini meliputi melakukan peng-*coding*-an perangkat, mengklasifikasikan bagian implementasi sesuai dengan deskripsi umum implementasi, lingkungan dan batasan implementasi. Implementasi pengembangan sistem ini dilaksanakan dengan menggunakan bahasa pemrograman java dan database My SQL. Mekanisme analisa penyusupan dilaksanakan dengan penerapan metode *Signature*, *Anomaly* dan *Hybrid Intrusion Detection*. Pada implementasi berisi juga tentang alasan pemilihan perangkat lunak yang digunakan yaitu *Java Security* beserta batasan implementasi dan lingkungan implementasi.

3.1.5 Pengujian

Setelah dilaksanakan implementasi, maka akan dilakukan pengujian terhadap implementasi yang telah dilaksanakan dan tinjauan ulang terhadap unjuk kerja sistem. Sistem ini digunakan pada Linux Fedora 10.

Pada pengujian sistem dilakukan pengujian terhadap perangkat lunak dan perangkat keras. Klasifikasi perangkat lunak dan keras dapat dilihat detail pada bab pengujian. Pengujian yang dilakukan terdiri dari dua bagian yaitu Pengujian *Link System* dan Pengujian Penyusupan Jaringan. Pengujian *Link System* berfungsi dalam menentukan kesesuaian *link* yang dimiliki *system* sehingga tidak ditemukan

kesalahan unjuk kerja sistem. Pengujian Penyusupan Jaringan berguna dalam melakukan evaluasi sistem dalam mewujudkan tujuan penelitian ini .Pengujian Penyusupan Jaringan terdiri atas dua bagian yaitu penyusupan *host* menggunakan *DoSHttp* dan penyusupan *network* menggunakan *Ping of Death.*

BAB IV

ANALISA DAN PERANCANGAN

Analisa merupakan hal yang sangat penting dan harus diperhatikan dalam membangun sistem. Analisa ialah suatu proses identifikasi permasalahan dari kumpulan data yang bernilai informasi dan bermanfaat pada pembangunan sistem. Setelah dilaksanakan analisa, maka langkah selanjutnya adalah perancangan terhadap sistem yang akan dibangun. Perancangan merupakan penentuan rincian hasil analisa sehingga dapat bersesuaian dengan analisa yang sebelumnya telah dilaksanakan

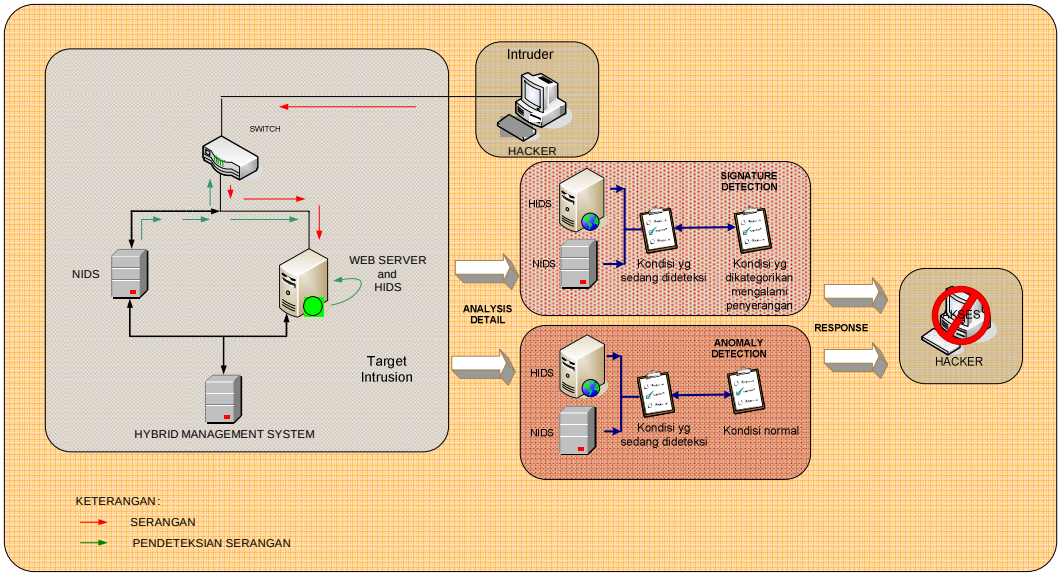
4.1 Deskripsi Umum Perangkat Lunak

Deskripsi umum merupakan pengambaran umum unjuk kerja atau perangkat lunak yang akan dibangun. Perangkat lunak yang akan dibangun adalah perangkat lunak yang memiliki kemampuan untuk pendeteksian terhadap penyusupan pada jaringan komputer baik secara pendekatan *host* maupun *network*. Perangkat ini juga memiliki kemampuan untuk menganalisa dengan metode *signature* dan *anomaly detection*.

Perangkat ini melakukan unjuk kerja yang terdiri dari tiga bagian yaitu *capture*, analisa dan respon. Pada unjuk kerja *capture*, perangkat melakukan *capture* terhadap *traffic* paket-paket di dalam sebuah jaringan. *Capture traffic* ini didasarkan pendekatan *host* dan *network*.Selanjutnya, hasil *capture* akan dianalisa dengan pendekatan metode *anomaly* dan *signature*. Pendekatan *signature* akan

menghasilkan *alert* (peringatan) adanya penyusupan, apabila kondisi *traffic* memiliki kesamaan dengan kondisi yang dikategorikan sebagai penyusupan, sedangkan pendekatan *anomaly*, *alert* akan dihasilkan, apabila kondisi *traffic* yang sedang dideteksi tidak memiliki kesamaan dengan kondisi normal *traffic*. Selanjutnya, *administrator* perangkat ini dapat memutuskan apakah melakukan respon terhadap paket *traffic* yang dikategorikan sebagai *alert*.

Berikut penggambaran umum dari perangkat lunak yang akan dibangun pada gambar 4.1



Gambar 4.1 Deskripsi Umum Perangkat Lunak

Rincian penjelasan deskripsi umum perangkat lunak, dapat dilihat pada uraian dibawah ini, sebagai berikut :

1. *Intruder* (Penyusup), melakukan penyusupan pada target operasi dengan sasaran utama yaitu *Web Server* yang dimiliki sebuah *network*

2. *Hybrid IDS Management System (HyIDS ManSys)* merupakan sistem utama yang memegang kendali terhadap keamanan *traffic network*. *HyIDS ManSys* memiliki beberapa tugas, yaitu:
    a. Menyimpan data kondisi *signature* dan *anomaly*
    b. Melakukan pendeteksian secara *real time* pada *resource host (Web Server)* dan *traffic* jaringan
    c. Menganalisa kondisi *traffic* jaringan dan *host* baik secara *signature* dan *anomaly detection*
    d. Memberikan peringatan (*alert*) terhadap kondisi yang dikategorikan sebagai penyusupan
    e. Melakukan *response* berupa *access blocking* terhadap penyusupan yang dilakukan *Intruder*

4.2 Analisa Perangkat Lunak

Analisa Perangkat Lunak bermanfaat sebagai penentu proses pengerjaan agar ditemukannya suatu pemecahan masalah. Terciptanya keruntunan suatu analisa perangkat lunak pada jalur yang benar merupakan dasar dilakukannya tahap ini.

Analisa perangkat lunak terdiri atas analisa kebutuhan data, analisa kebutuhan fungsi, analisa metode yang akan digunakan pada Perangkat Lunak, analisa fitur yang akan dibangun pada perangkat lunak, dan analisa fungsional.

4.2.1 Analisa Kebutuhan Data

Analisa kebutuhan data yang terdapat pada perangkat lunak ini, ialah :

1. Inisialisai data awal perangkat lunak berdasarkan metode yang digunakan, terbagi atas dua hal, sebagai berikut :
    a. *Signature* Data, berisi data yang dikategorikan sebagai penyusupan pada *host* dan *traffic* jaringan. Berdasarkan metode yang digunakan data ini dapat berupa;
        i. *Host* memiliki data berupa *ip address host, ip address* yang berhak untuk mengakses *host*, *event* yang dilakukan *host*
        ii. *Traffic* memiliki data berupa *ip address* tujuan, *ip address* asal, *port*, protokol
    b. Data *Anomaly* berisi data kondisi normal suatu *host* dan *traffic* jaringan. Detail data memiliki kesamaan dengan Data *signature*.
2. Menangkap (*capture*) atau proses *sniffer* terhadap data kondisi pada resource *host* dan *traffic* jaringan, ketika dilakukan pengaktifan *HyIDS ManSys* baik pada *host* maupun *traffic* dan keduanya secara bersamaan. Secara umum, pencatatan juga dilakukan pada data tanggal dan kondisi proses *sniffer*. Berikut penjelasan lebih lanjut mengenai proses *sniffer* berdasarkan data kondisi pada *host* dan *traffic* yaitu :
    a. *Host* memiliki data berupa *ip address host, ip address* yang berhak untuk mengakses *host*, *event* yang dilakukan *host*
    b. *Traffic* memiliki data berupa *ip address* tujuan, *ip address* asal, *port*, protokol

4.2.2 Analisa Kebutuhan Fungsi

Perangkat lunak ini memiliki beberapa fungsi yang bisa dimanfaatkan oleh user agar dapat menghasilkan output yang maksimal dan berjalan sebagaimana mestinya. Berikut rincian analisa kebutuhan fungsi bagi user, yaitu :

1. Fungsi aktivasi *Network Intrusion Detection* dan *Host Intrusion Detection*
2. Fungsi input *network signature* dan *host signature*
3. Fungsi *capture* kondisi normal yang berguna dalam proses analisa *Anomaly Detection*
4. Fungsi Sniffing atau proses penangkapan kondisi *Host* dan *Network*
5. Fungsi proses analisa secara *signature* dan *anomaly detection* baik pada *host* dan *network*
6. Fungsi penyimpanan *alert* atau peringatan dan *response*

4.2.3 Analisa Metode yang akan digunakan pada Perangkat Lunak

Analisa metode yang dilakukan oleh *Hybrid Intrusion Detection Management System* meliputi dua bagian, sebagai berikut :

1. *Host Intrusion Detection System (HIDS)*, pada bagian ini terdapat dua metode pendeteksian yang akan dilakukan, yaitu :

a. *Signature Detection*, merupakan proses penganalisaan suatu kondisi *host* dengan membandingkan kondisi yang sedang dideteksi dengan *signature rules* yang telah diinputkan sebelumnya. *Signature rules* berisi *rules* yang dikategorikan sebagai penyusupan.

b. *Anomaly detection* ialah proses penganalisaan suatu kondisi *host* dengan membandingkan kondisi normal *host* dengan kondisi yang sedang

terdeteksi, selanjutnya akan dikategorikan sebagai penyusupan apabila terjadi perbedaan dari kedua kondisi.

2. *Network Intrusion Detection System (NIDS),* pada bagian ini terdapat dua metode yang akan dilaksanakan, yaitu :
   a. *Signature Detection*, merupakan proses penganalisaan suatu kondisi *traffic* dengan membandingkan kondisi yang sedang dideteksi dengan *signature rules* yang telah diinputkan sebelumnya. *Signature rules* berisi *rules* yang dikategorikan sebagai penyusupan.
   b. *Anomaly detection* ialah proses penganalisaan suatu kondisi *traffic* dengan membandingkan kondisi normal *traffic* dengan kondisi yang sedang terdeteksi, selanjutnya akan dikategorikan sebagai penyusupan apabila terjadi perbedaan dari kedua kondisi.

Indikator-indikator pada proses analisa oleh kedua bagian ini dapat terlihat pada tabel 4.1 :

Tabel 4.1 Indikator-indikator proses analisa metode perangkat lunak

HIDS *(Host Intrusion Detection System)*		*NIDS* *(Network Intrusion detection System)*	
Meode *Signature*	Metode *Anomaly*	Meode *Signature*	Metode *Anomaly*
IP Address	*IP Address*	*IP Address*	*IP Address*
Port	*Port*	*Port*	*Port*

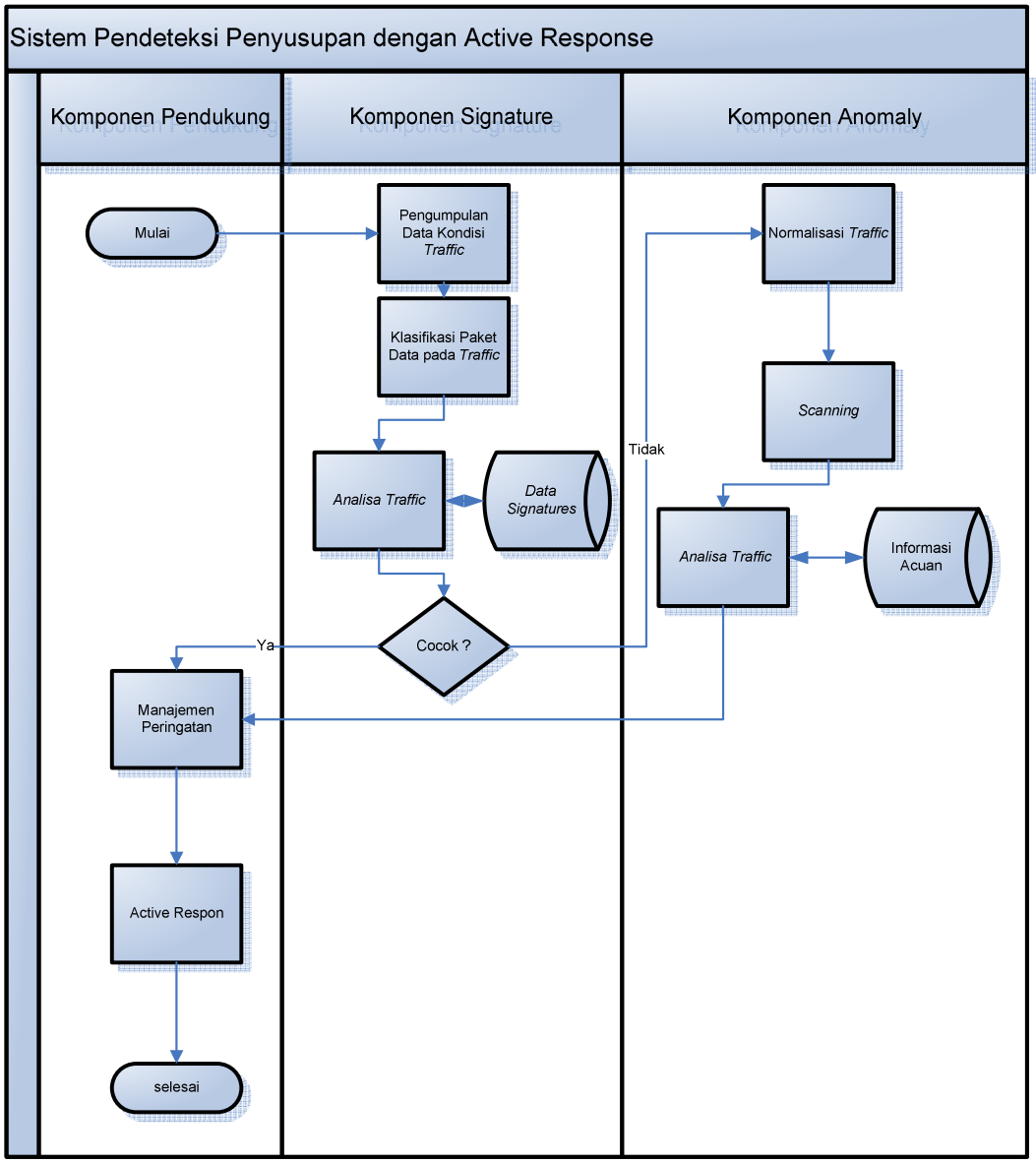
Keterangan :

Secara sekilas, terlihat kesamaan indikator masing-masing metode. Hal ini didasari oleh pembatasan masalah dalam pembangunan perangkat ini. Perbedaan antara keempat metode ini bisa terlihat disaat diinputkannya *field* dan berlangsungnya proses metode-metode analisa *traffic* dan *host*. Hal ini bisa tergambar pada penjelasan, sebagai berikut :

1. *Signature* pendekatan *host* berisi field kondisi *host* yang dikategorikan penyusupan. Pada proses analisa, sebuah kondisi pendeteksian yang diindikasi sebagai penyusupan apabila memiliki kesamaan dengan *field signature host*
2. Metode *Anomaly* dengan pendekatan *host* berisi field kondisi *host* yang dikategorikan sebagai kondisi normal. Pada proses analisa, sebuah kondisi pendeteksian yang diindikasi sebagai penyusupan apabila memiliki perbedaan dengan *field* kondisi normal *host*.
3. *Signature* pendekatan *network* berisi field kondisi *network* keseluruhan yang dikategorikan penyusupan. Pada proses analisa, sebuah kondisi pendeteksian yang diindikasi sebagai penyusupan apabila memiliki kesamaan dengan *field signature network*.
4. Metode *Anomaly* dengan pendekatan *network* berisi *field* kondisi *network* keseluruhan yang dikategorikan sebagai kondisi normal. Pada proses analisa, sebuah kondisi pendeteksian yang diindikasi sebagai penyusupan apabila memiliki perbedaan dengan *field* kondisi normal *network*.

Berikut proses pendeteksian dapat tergambar pada *flow chart* dibawah ini :

Tabel 4.2 Metode *Signature* dan *Anomaly*



4.2.4 Analisa Fitur yang akan dibangun pada Perangkat Lunak

Fitur yang dikembangkan pada perngkat lunak ini, ialah :

a. *Sniffer mode*, untuk melihat paket lewat di *network*

b. Paket *Logger mode*, untuk mencatat semua paket yang lewat di *network* dan akan dilakukan penganalisaan di masa mendatang

c. *Network Intrusion Detection mode*, berfungsi untuk mendeteksi serangan yang melalui jaringan

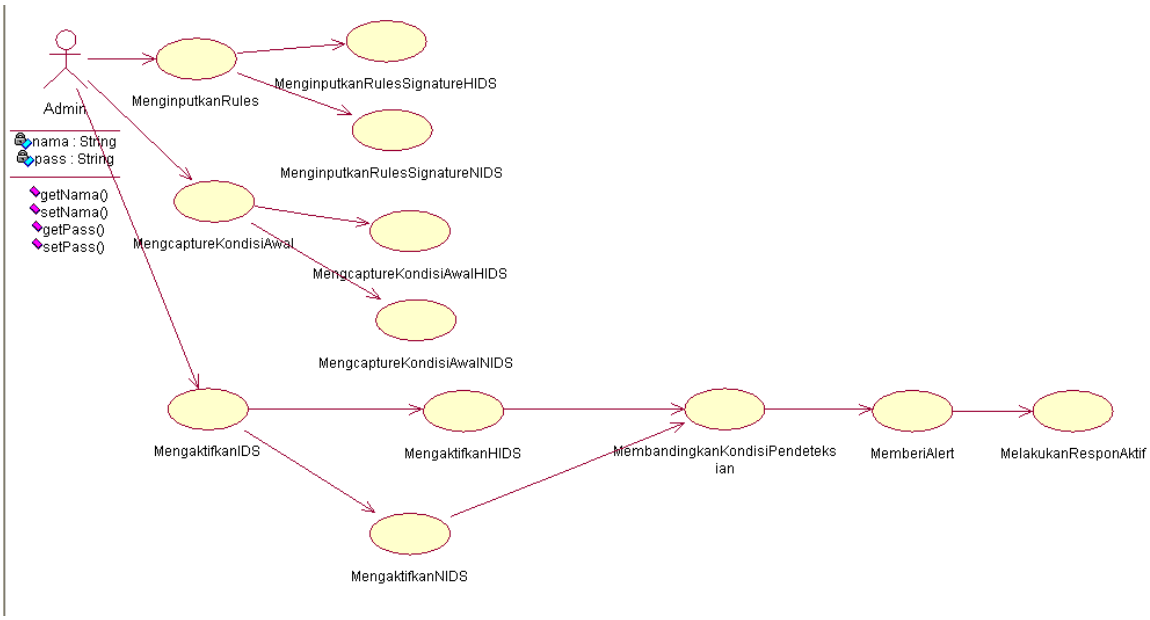
d. *Host Intrusion Detection mode*, berfungsi untuk mendsteksi serangan yang terjadi pada suatu *host*

4.2.5 Analisa Fungsional

Analisa fungsional yang digunakan didasari pada *Unified Managed language (UML)*. Analisa fungsional ini terdari pembahasan *Use Case Diagram, Class Diagram* dan *Deployment Diagram.*

4.2.5.1 Use Case Diagram

Fungsionalitas antara aktor dan *use case* dapat digambarkan sebagai *use case diagram.* Admin merupakan satu-satunya aktor yang terlibat pada perangkat lunak. Berikut *Use Case Diagram* dari perangkat lunak ini :



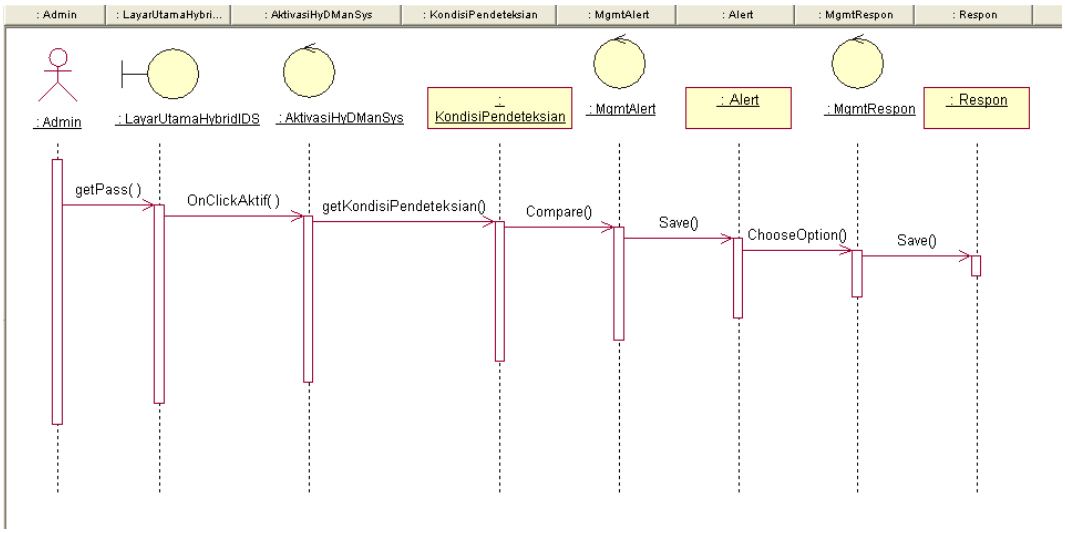
Gambar 4.2 *Use Case Diagram*

Tabel 4.3 Keterangan Use Case Diagram

No	*Use Case*	Keterangan
1	*MenginputkanRules*	Proses untuk memberikan informasi mengenai kondisi penyusupan
2	*MengcaptureKondisiAwal*	Proses yang mengidentifikasi variabel pembandingan pada *anomaly detection*
3	*MengaktifkanIDS*	Proses pengaktifan IDS
4	Menginputkan*RulesSignature*HIDS	Proses untuk memberikan informasi mengenai kondisi penyusupan pada *host*
5	Menginputkan*RulesSignature*NIDS	Proses untuk memberikan informasi mengenai kondisi penyusupan pada *traffic*
6	Meng*capture*KondisiAwalHIDS	Proses yang mengidentifikasi variabel pembandingan pada *anomaly detection* pada *host*
7	Meng*capture*KondisiAwalNIDS	Proses yang mengidentifikasi variabel pembandingan pada *anomaly detection*
8	MengaktifkanHIDS	Proses pengaktifan HIDS
9	MengaktifkanNIDS	Proses pengaktifan NIDS
10	MembandingkanKondisiPendeteksian	Membandingkan kondisi pendeteksian dengan *rules*
11	TidakTerdeteksiPenyusupan	Tidak ditemukannya kondisi yang dikategorikan sebagai penyusupan
12	TerdeteksiPenyusupan	Ditemukannya kondisi yang dikategorikan sebagai penyusupan
13	Memberi*Alert*	Peringatan jika ditemukan adanya penyusupan
14	MelakukanResponAktif	Respon aktif terhadap peringatan yang dikeluarkan

4.2.5.2 *Sequence Diagram*

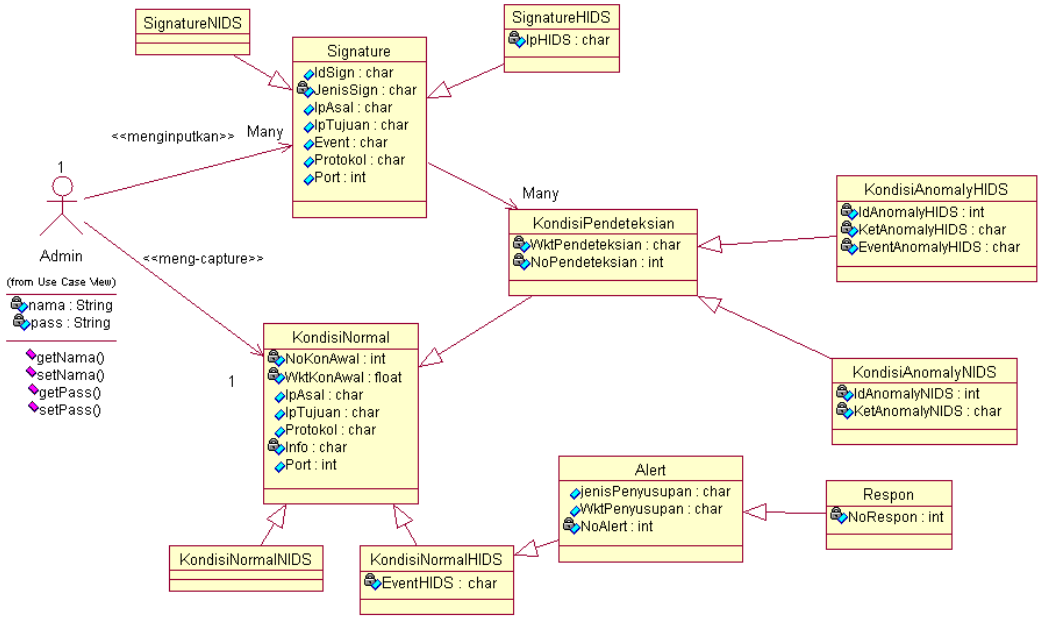
Sequence Diagram berfungsi dalam visualisasi interaksi antar objek pada perangkat. Berikut gambar *sequence diagram* pada perangkat ini :



Gambar 4.3 *Sequence Diagram*

4.2.5.3 *Class Diagram*

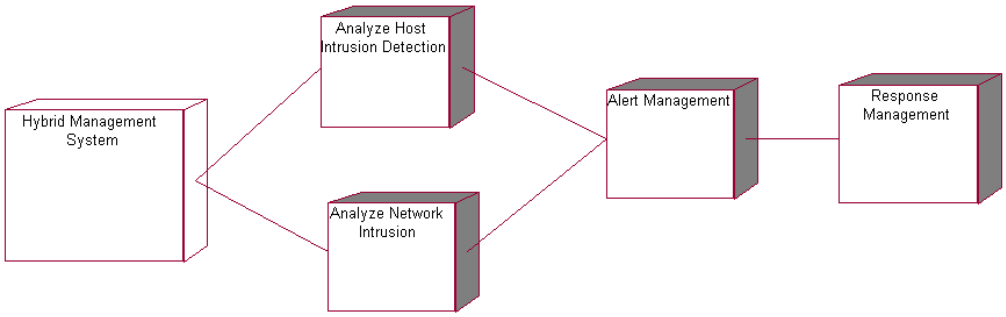
Class Diagram bermanfaat dalam menggambat *class* baik secara struktur, deskripsi maupun relational atau hubungan antar *class*. *Class Diagram* juga dapat memberikan informasi mengenai analisa data dalam perangkat lunak, seperti yang terlihat pada gambar dibawah ini.



Gambar 4.4 *Class Diagram*

4.2.5.4 *Deployment Diagram*

Setiap komponen dari perangkat lunak ini akan dikembangkan dalam infrastrktur sehingga terlihatlah dimana letak komponen-komponen secara detail. *Deployment Diagram* dapat terlihat pada gambar dibawah ini :



Gambar 4.5 *Deployment Diagram*

Tabel 4.4 Keterangan *Deployment Diagram*

No	*Deployment Diagram*	Keterangan
1.	*Hybrid Management System*	Komponen utama dalam pengelolaan IDS
2.	*Analyze Host Intrusion Detection*	Komponen yang berfungsi dalam penganalisaan kondisi *host*
3.	*Analyze Network Intrusion Detection*	Komponen yang berfungsi dalam penganalisaan kondisi *traffic*
4.	*Alert Management*	Komponen yang mengelola *alert* pada IDS
5.	*Reseponse Management*	Komponen yang mengelola respon aktif pada IDS

4.3 Perancangan Perangkat Lunak

Perancangan ialah tahap yang dilaksanakan setelah dilakukan analisa pada perangkat lunak. Perancangan ini berguna agar proses implementasi yang dilakukan dapat konsisten dan sesuai dengan analisa yang dilakukan. Perancangan yang dilakukan meliputi perancangan *database*, struktur menu dan tampilan.

4.3.1 Perancangan *Database*

Database merupakan bagian yang sangat berperan penting dalam perangkat ini. Di dalam database lah disimpan *rules* yang berkaitan dengan unjuk kerja dari *Hybrid IDS* ini, serta penyimpanan hasil-hasil kegiatan dalam pendeteksian penyusupan jaringan, seperti kondisi pendeteksian, *management alert* dan *management response*.

Perancangan *Database* yang dilaksanakan pada perangkat lunak ini berdasarkan tabel data yang digunakan. Ada sembilan tabel yang terdapat pada perangkat lunak ini. Namun, pada bab ini hanya ditampilkan beberapa dari keseluruhan tabel, dan untuk tabel yang lain dapat dilihat pada lampiran. Tabel tersebut, sebagai berikut :

1. Tabel *User*

   Deskripsi : Berisi data-data yang digunakan untuk menampung data pengguna yang akan menggunakan perangkat lunak ini

   Primary key : *Username* dan *Password*

Tabel 4.5 Struktur Tabel *User*

No	Nama *Field*	Tipe Data	Keterangan
1	*Username**	Text (25)	*Username* untuk login ke dalam perangkat lunak (*Primary Key*)
2	*Password**	Text (10)	*Password* untuk login ke dalam perangkat lunak (*Primary Key*)

2. Tabel *SignatureNIDS*

Deskripsi : Berisi data-data yang digunakan untuk menampung data *rules SignatureNIDS*

Primary key : *IdSign*

Tabel 4.6 Struktur Tabel *SignatureNIDS*

No	Nama *Field*	Tipe Data	Keterangan
1	*IdSign* *	Char(10)	*IdSign* sebagai ID unik untuk membedakan setiap rule
2	*JenisSign*	Char (10)	*JenisSign* digunakan untuk menidentifikasi jenis *Signature* (NIDS atau HIDS)
3	*IpAsal*	Char(20)	*IpAsal* merupakan Ip Address pengirim event pada suatu *host*
4	*IpTujuan*	Char(20)	*IpTujuan* ialah IP Address *host* yang melakukan event
5	*Protokol*	Char(10)	*Protokol* adalah spesifikasi jenis protocol yang digunakan (misalnya icmp,udp,tcp)
6	*Port*	Char(10)	*Port* ialah spesfifiksi jenis port yang digunakan (misalnya 8080,80,52)

3. Tabel *SignatureHIDS*

Deskripsi : Berisi data-data yang digunakan untuk menampung data *rules SignatureNIDS*

Primary key : *IdSign dan IpHIDS*

Tabel 4.7 Struktur Tabel *SignatureHIDS*

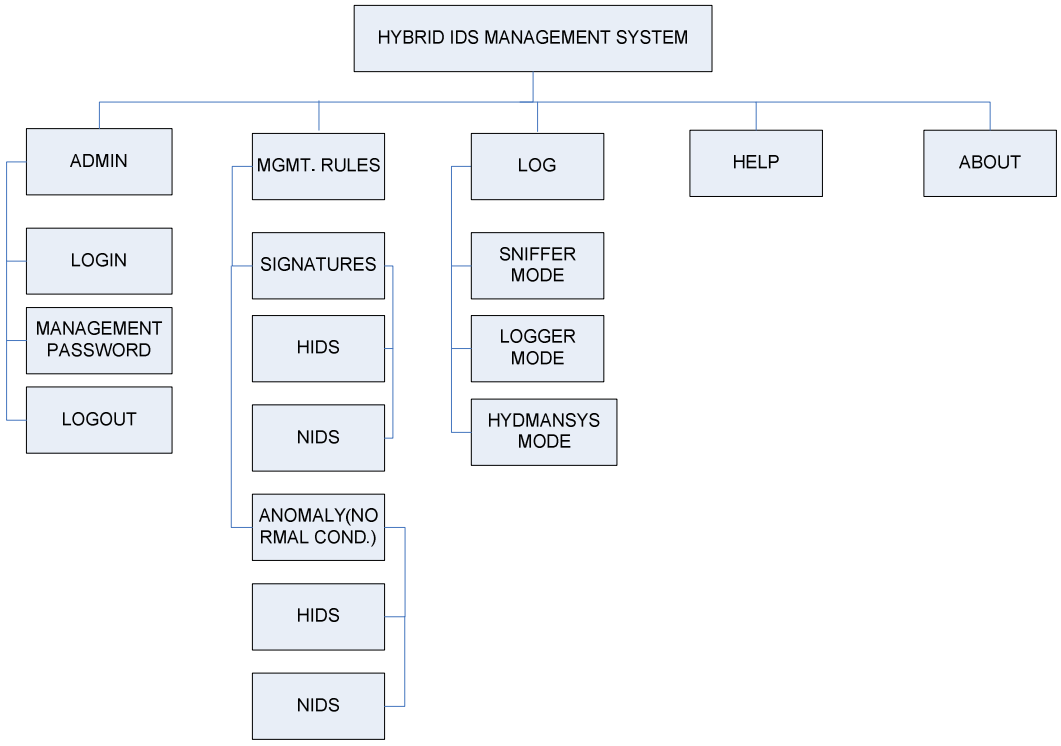
No	Nama *Field*	Tipe Data	Keterangan
1	*IdSign* *	Char(10)	*IdSign* sebagai ID unik untuk membedakan setiap rule
2	*JenisSign*	Char (10)	*JenisSign* digunakan untuk menidentifikasi jenis *Signature* (NIDS atau HIDS)
3	*IpAsal*	Char(20)	*IpAsal* merupakan Ip Address pengirim event pada suatu *host*
4	*IpTujuan*	Char(20)	*IpTujuan* ialah IP Address *host* yang melakukan event
5	*Protokol*	Char(10)	*Protokol* adalah spesifikasi jenis protocol yang digunakan (misalnya icmp,udp,tcp)
6	*Port*	Char(10)	*Port* ialah spesfifiksi jenis port yang digunakan (misalnya 8080,80,52)

4.3.2 Perancangan Tampilan Sistem

Perancangan terhadap tampilan yang akan dibangun merupakan hal yang sangat penting dalam terciptanya kekonsistenan implementasi. Perancangan tampilan ini terdiri atas dua bagian yaitu struktur menu dan *interface*.

4.3.2.1 Perancangan Struktur Menu

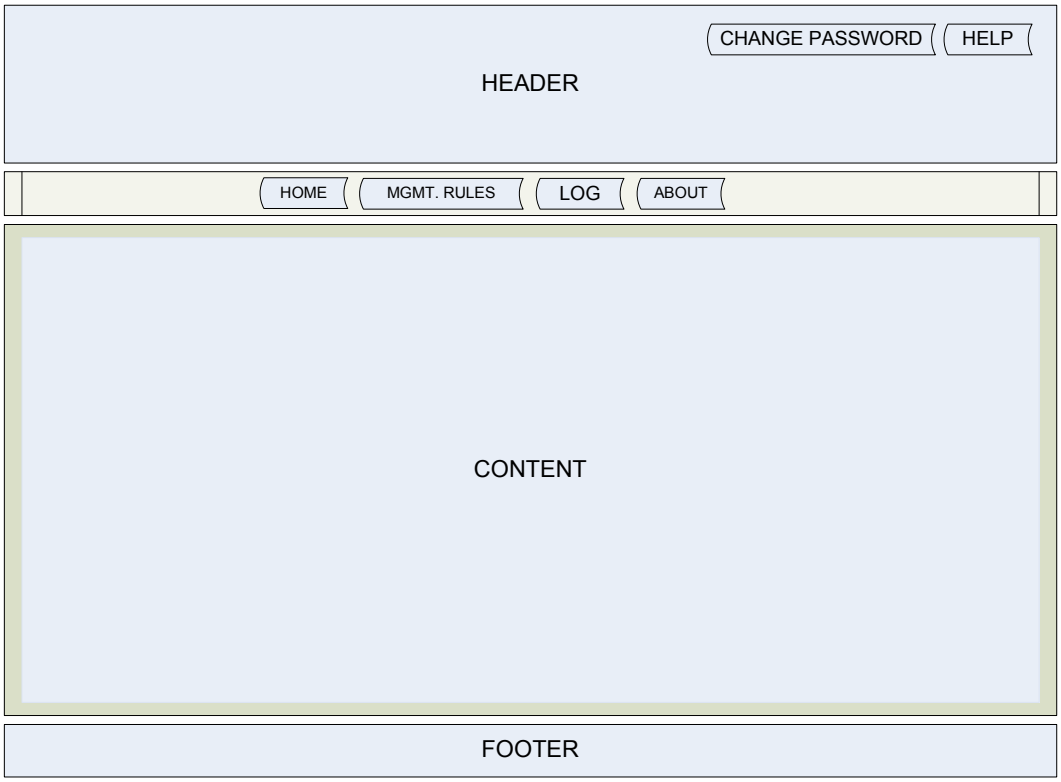
Perancangan Struktur Menu bermanfaat dalam mengawali perancangan tampilan. Struktur menu dapat terlihat pada gambar dibawah ini :



Gambar 4.6 Struktur Menu

4.3.2.2 Perancangan *Interface*

Berdasarkan struktur menu yang telah dirancang, berikut gambar dari tampilan yang akan dibuat :



Gambar 4.7 Tampilan Perangkat Lunak

4.3.3 Perancangan Prosedural

Perancangan prosedural yang dilaksanakan pada pengembangan perangkat lunak ini dapat terlihat *pseudo code* algoritma unjuk kerja utama untuk melakukan proses pendeteksian penyusupan. Perancangan prosedural dari perangkat ini, sebagai berikut :

1. Algoritma *pseudo code* dari *class* Deteksi

```
Public class Deteksi()
DEKRALASI
idDeteksi,port : integer
tgl,jam,ipAsal,ipTujuan,protocol,event : string
ALGORITMA
Method  :    getidDeteksi
             setidDeteksi
             getipAsal
             setipAsal
             getipTujuan
             setipTujuan
             getJam
             setJam
             getPort
             setPort
             getTgl
             setTgl
             getEvent
             setEvent
```

2. Algoritma *pseudo code* dari *class Manager* Deteksi

```
Public class ManagerDeteksi()
DEKRALASI
idDeteksi,port : integer
tgl,jam,ipAsal,ipTujuan,protocol,event : string
ALGORITMA
Method :   simpanPacketTCP
           simpanPacketICMP
           simpanPacketUDP
           getAllDeteksiHost
           getAllDeteksiNids
simpanPacketTCP( obj : Deteksi){
     sql          ←          “INSERT          INTO
     konpkhost(tgl,jam,IpAsal,IpTujuan,Protokol,Port,
     eventhids)                                VALUES
     (current_date,current_time,?,?,?,?,?)
     try
     {    DBKoneksi db ← new DBKoneksi()
          PreparedStatement pst ← db.getKoneksi()
          setString(obj.getIpAsal())
          setString(obj.getIpTujuan())
          setString(obj.getProtokol())
          setString(obj.getPort())
          setString(obj.getEvent())
          pst.executeUpdate()
          db.closeKoneksi()
```

```
      } catch (SQlException){
      Throw RuntimeEception()}

simpanPacketICMP(     obj : Deteksi){
      sql           ←           “INSERT           INTO
      konpknids(tgl,jam,IpAsal,IpTujuan,Protokol,Port)
      VALUES (current_date,current_time,?,?,?,?)
      try
      {     DBKoneksi db ← new DBKoneksi()
            PreparedStatement pst ← db.getKoneksi()
            setString(obj.getIpAsal())
            setString(obj.getIpTujuan())
            setString(obj.getProtokol())
            setString(obj.getPort())
            pst.executeUpdate()
            db.closeKoneksi()
      } catch (SQlException){
      Throw RuntimeEception()}

simpanPacketUDP( obj : Deteksi){
      sql           ←           “INSERT           INTO
      konpknids(tgl,jam,IpAsal,IpTujuan,Protokol,Port)
      VALUES (current_date,current_time,?,?,?,?)
      try
      {     DBKoneksi db ← new DBKoneksi()
            PreparedStatement pst ← db.getKoneksi()
```

```
                setString(obj.getIpAsal())
                setString(obj.getIpTujuan())
                setString(obj.getProtokol())
                setString(obj.getPort())
                pst.executeUpdate()
                db.closeKoneksi()
          } catch (SQlException){
          Throw RuntimeEception()}

getAllDeteksiHost()
{   List<Deteksi> ls ← new ArrayList<Deteksi>
    Sql = “SELECT * FROM konpkhost ORDER BY tgl, jam”
    Try
          {     DBKoneksi db ← new DBKoneksi()
                Statement st ← db.getKoneksi()
                Resultset rs ← st.executeQuery(sql)
                Deteksi obj
                While (rs.next()) {
                      Obj ← new DBKoneksi();
                      obj.setIdDeteksi(rs.getIdPdtksianHost)
                      obj.setTgl(rs.getTgl)
                      obj.setJam(rs.getJam)
                      obj.setPort(rs.getPort)
                      obj.setProtokol(rs.getProtokol)
                      obj.setIpAsal(rs.getIpAsal)
                      obj.setIpTujuan(rs.getIpTujuan)
```

```
                    obj.setEvent(rs.getEventHIDS)
                    ls.add(obj)
                endwhile
            db.closeKoneksi()

        catch (SQLException se)
        {  se.printStackTrace()
            throw new RuntimeException(se)}
        return ls }

getAllDeteksiNids()
{
        List<Deteksi> ls ← new ArrayList<Deteksi>()
        sql ← "SELECT * FROM konpknids ORDER BY tgl,jam"
        try
        {
            DBKoneksi db ← new DBKoneksi()
            Statement st ← db.getKoneksi()
            ResultSet rs ← st.executeQuery(sql);
            Deteksi obj
            while (rs.next())
                obj = new Deteksi()
                obj.setIdDeteksi(rs.getIdPdtksianNIDS)
                obj.setTgl(rs.getTgl)
                obj.setJam(rs.getJam)
                obj.setPort(rs.getPort)
```

```
            obj.setProtokol(rs.getProtokol)
            obj.setIpAsal(rs.getIpAsal)
            obj.setIpTujuan(rs.getIpTujuan)
            ls.add(obj)
        endwhile
         db.closeKoneksi() }

    catch (SQLException se)
    {   se.printStackTrace()
        throw new RuntimeException(se)}
    return ls  }
```

3. Algoritma *pseudo code* dari *class Alert*

```
Public class Alert()
DEKRALASI
idAlert,idPtkdian : integer
jenisPenyusupan : string
ALGORITMA
Method     :    getDeteksi
                setDeteksi
                getidAlert
                setidAlert
                getidPdtksian
                setidPdtksian
                getJenisPenyusupan
                setJenisPenyusupan
```

4. Algoritma *pseudo code* dari *class Manager Alert*

```
public class ManagerAlert
ALGORITMA
Method :   getDeteksiNetwork
           getDeteksiNetworkHyd
           getDeteksi
           getDeteksiHostHyd
           simpanAlert
           compareSignatureHIDS
           compareSignatureNIDS
           compareKonNorm
           compareKonNIDS
           getAlert

public List<Deteksi> getDeteksiNetwork(String
tgl_awal, String tgl_akhir)
    {
        List<Deteksi> ls ← new ArrayList<Deteksi>()
        sql = "SELECT * FROM konpknids WHERE tgl >=
              tgl_awal + "' AND tgl <='" + tgl_akhir"
    try
        { DBKoneksi db ← new DBKoneksi()
            Statement st ← db.getKoneksi()
            ResultSet rs ← st.executeQuery(sql)
            Deteksi obj
            while (rs.next())
```

```
                obj = new Deteksi()
                obj.setIdDeteksi(rs.getIdPdtksianNids)
                obj.setPort(rs.getPort)
                obj.setProtokol(rs.getProtokol)
                obj.setIpAsal(rs.getIpAsal)
                obj.setIpTujuan(rs.getIpTujuan)
                obj.setTgl(rs.getTgl)
                obj.setJam(rs.getJam)
                ls.add(obj)
             endwhile
            db.closeKoneksi()
         }  catch (SQLException se)
            { se.printStackTrace()
              throw new RuntimeException(se) }
return ls  }

 public List<Deteksi> getDeteksiNetworkHyd()
    {   List<Deteksi> ls ← new ArrayList<Deteksi>()
        sql ← "SELECT * FROM konpknids "
    try
        {   DBKoneksi db ← new DBKoneksi()
            Statement st ← db.getKoneksi()
            ResultSet rs ← st.executeQuery(sql);
            Deteksi obj
            while (rs.next())
                obj ← new Deteksi()
```

```
                obj.setIdDeteksi(rs.getIdPdtksianNids)
                obj.setPort(rs.getPort)
                obj.setProtokol(rs.getProtokol)
                obj.setIpAsal(rs.getIpAsal)
                obj.setIpTujuan(rs.getIpTujuan)
                obj.setTgl(rs.getTgl)
                obj.setJam(rs.getJam)
            endwhile
                ls.add(obj)       }
            db.closeKoneksi()  }
        catch (SQLException se)
        {   se.printStackTrace()
            throw new RuntimeException(se)  }
   return ls    }

public List<Deteksi> getDeteksi(String tgl_awal, String
tgl_akhir)
     {
         List<Deteksi> ls = new ArrayList<Deteksi>();
         sql = "SELECT * FROM konpkhost WHERE tgl >= '"
             tgl_awal + AND tgl <='" + tgl_akhir "'"
     try
         {
             DBKoneksi db ← new DBKoneksi()
             Statement st ← db.getKoneksi()
             ResultSet rs ← st.executeQuery(sql)
```

```
            Deteksi obj
            while (rs.next())
               obj ← new Deteksi()
               obj.setIdDeteksi(rs.getIdPdtksianHost)
               obj.setPort(rs.getPort)
               obj.setProtokol(rs.getProtokol)
               obj.setIpAsal(rs.getIpAsal)
               obj.setIpTujuan(rs.getIpTujuan)
               obj.setTgl(rs.getTgl)
               obj.setJam(rs.getJam)
            endwhile
                ls.add(obj)            }
            db.closeKoneksi()        }
        catch (SQLException se)
        {   se.printStackTrace()
            throw new RuntimeException(se) }
 return ls    }

public List<Deteksi> getDeteksiHostHyd()
   {  List<Deteksi> ls ← new ArrayList<Deteksi>()
       sql = "SELECT * FROM konpkhost"
   try
       {  DBKoneksi db ← new DBKoneksi()
           Statement st ← db.getKoneksi()
           ResultSet rs ← st.executeQuery(sql)
           Deteksi obj
```

```
            while (rs.next())
                obj ← new Deteksi()
                obj.setIdDeteksi(rs.getIdPdtksianHost)
                obj.setPort(rs.getPort)
                obj.setProtokol(rs.getProtokol)
                obj.setIpAsal(rs.getIpAsal)
                obj.setIpTujuan(rs.getIpTujuan)
                obj.setEvent(rs.geteventHIDS)
                obj.setTgl(rs.getTgl)
                obj.setJam(rs.getJam)
            endwhile
                ls.add(obj) }
            db.closeKoneksi()        }
        catch (SQLException se)
        {   se.printStackTrace()
            throw new RuntimeException(se)  }
  return ls  }

public simpanAlert( Alert obj)
    {   sql = "INSERT INTO
              alert(JenisPenyusupan,IdPdtksian) VALUES
              (?,?)"
     try
     {    DBKoneksi db ← new DBKoneksi()
          PreparedStatement pst ←
         db.getKoneksi().prepareStatement(sql)
```

```
            pst.setString(1, obj.getJenisPenyusupan())
            pst.setInt(2, obj.getIdPdtksian())
            pst.executeUpdate()
            db.closeKoneksi()

    catch (SQLException se)
          se.printStackTrace()
          throw new RuntimeException(se)

public boolean compareSignatureHIDS(Deteksi det,
SignatureHIDS sign )
          status  ← false
          cek_ipasal ← false;
          cek_iptujuan ← false;
          cek_port ← false;
          cek_protokol ← false

      if ( det.getIpAsal()==(sign.getIpAsal()))
              cek_ipasal ← true
      if ( det.getIpTujuan()==(sign.getIpTujuan()))
           cek_iptujuan ← true
     if ( det.getProtokol()==(sign.getProtokol()))
          cek_protokol ← true
      if (det.getPort() == sign.getPort())
          cek_port ← true
```

```
  if ( cek_ipasal && cek_iptujuan && cek_port &&
  cek_protokol)
       status ← true
return status

public boolean compareSignatureNIDS(Deteksi det,
SignatureNIDS sign )
         status  ← false
         cek_ipasal ← false
         cek_iptujuan ← false
         cek_port ← false
         cek_protokol ← false

if ( det.getIpAsal()==(sign.getIpAsal()))
 cek_ipasal ← true
if ( det.getIpTujuan()==(sign.getIpTujuan()))
    cek_iptujuan←true
if ( det.getProtokol()==(sign.getProtokol()))
    cek_protokol ← true
if (det.getPort() == sign.getPort())
    cek_port ← true
if ( cek_ipasal && cek_iptujuan && cek_port &&
cek_protokol)
  status ← true
 return status
```

```
public boolean compareKonNorm(Deteksi det,
KondisiNormHIDS konhost )
       status  ← false
       cek_ipasal ← false
       cek_iptujuan ← false
       cek_port ← false
       cek_protokol ← false

   if (det.getIpAsal()==(konhost.getIpAsal()))
       cek_ipasal ← true
   if ( det.getIpTujuan()== konhost.getIpTujuan()))
        cek_iptujuan ← true
   if ( det.getProtokol()==(konhost.getProtokol()))
        cek_protokol ← true
   if (det.getPort() == konhost.getPort())
        cek_port ← true
   if ( cek_ipasal && cek_iptujuan && cek_port &&
  cek_protokol)
        status ← true
   return status

  public boolean compareKonNIDS(Deteksi det,
  KondisiNormNIDS konhost )
    {    status  ← false
         cek_ipasal ← false
```

```
        cek_iptujuan ← false
        cek_port ← false
        cek_protokol ← false

   if (det.getIpAsal()==(konhost.getIpAsal()))
       cek_ipasal ← true
    if(det.getIpTujuan()==(konhost.getIpTujuan()))
       cek_iptujuan ← true
    if (det.getProtokol()==(konhost.getProtokol()))
       cek_protokol = true
   if (det.getPort() == konhost.getPort())
       cek_port = true
  if ( cek_ipasal && cek_iptujuan && cek_port &&
cek_protokol)
        status ← true
 return status

public List<Alert> getAlert(String jenis)
     {   List<Alert> ls ← new ArrayList<Alert>()
         String tabel ←""
         String kriteria ←""
      if (jenis == ("HOST"))
           tabel←" konpkhost "
           kriteria ← " b.Idpdtksianhost"
      else if (jenis==("NETWORK"))
```

```
      tabel ← "konpknids"
      kriteria ← " b.IdpdtksianNids"
     endif
       sql = "SELECT a.*,b.* FROM alert a INNER JOIN "
              tabel + " b ON a.IdPdtksian="+kriteria
   try
       {   DBKoneksi db ← new DBKoneksi()
           Statement st ← db.getKoneksi()
           ResultSet rs ←st.executeQuery(sql)
           Deteksi det
           Alert obj
           while (rs.next())
           {   obj ← new Alert()
               obj.setIdAlert( rs.getInt("IdAlert"))
               obj.setJenisPenyusupan(jenis)
                det = new Deteksi()
               det.setPort(rs.getPort)
               det.setProtokol(rs.getProtokol)
               det.setIpAsal(rs.getIpAsal)
               det.setIpTujuan(rs.getIpTujuan)

if ( jenis ==("HOST"))
     get.setIdDeteksi(rs.getIdPdtksianHost"))
      det.setEvent(rs.geteventHIDS)
else if (jenis ==("NETWORK"))
          det.setIdDeteksi(rs.getIdPdtksianNids)
```

```
  endif
            det.setTgl(rs.getTgl)
            det.setJam(rs.getJam)
            obj.setDeteksi(det)
   ls.add(obj)            }
             db.closeKoneksi()        }
  catch (SQLException se)
         {   se.printStackTrace()
             throw new RuntimeException(se)  }
return ls    }

public List<Alert> getAlert( String[] arr , String
jenis)
     {   List<Alert> ls ← new ArrayList<Alert>()
         String tabel ← ""
         String kriteria ← ""
      if (jenis==("HOST"))
          tabel←" konpkhost "
          kriteria ← " b.Idpdtksianhost and a.IdAlert="
      else if (jenis==("NETWORK"))
           tabel←" konpknids"
          kriteria ← " b.IdpdtksianNids and a.IdAlert="
       endif
     try
         {   DBKoneksi db ← new DBKoneksi()
```

```
            Statement st ← db.getKoneksi()
            Deteksi det
            Alert obj
            ResultSet rs
            for (int i = 0 ; i < arr.length ; i++)
              String sql = "SELECT a.*,b.* FROM alert
                           a INNER JOIN " +tabel + " b
                           ON a.IdPdtksian="+kriteria
                           + arr[i]
              rs ← st.executeQuery(sql)
            endfor
            while (rs.next())
               obj = new Alert()
               obj.setIdAlert( rs.getIdAlert)
               obj.setJenisPenyusupan(jenis)
               det = new Deteksi()
               det.setPort(rs.getPort)
               det.setProtokol(rs.getProtokol)
               det.setIpAsal(rs.getIpAsal)
               det.setIpTujuan(rs.getIpTujuan)

if ( jenis == ("HOST"))
      det.setIdDeteksi(rs.getIdPdtksianHost)
       det.setEvent(rs.getEventHIDS)
  else if (jenis==("NETWORK"))
       det.setIdDeteksi(rs.getIdPdtksianNids"))
```

```
      det.setTgl(rs.getTtgl)
      det.setJam(rs.getJam)
      obj.setDeteksi(det)
  ls.add(obj)
  endif
  endif
 db.closeKoneksi()       }
 catch (SQLException se)
          {   se.printStackTrace()
              throw new RuntimeException(se) }
return ls
```

BAB V

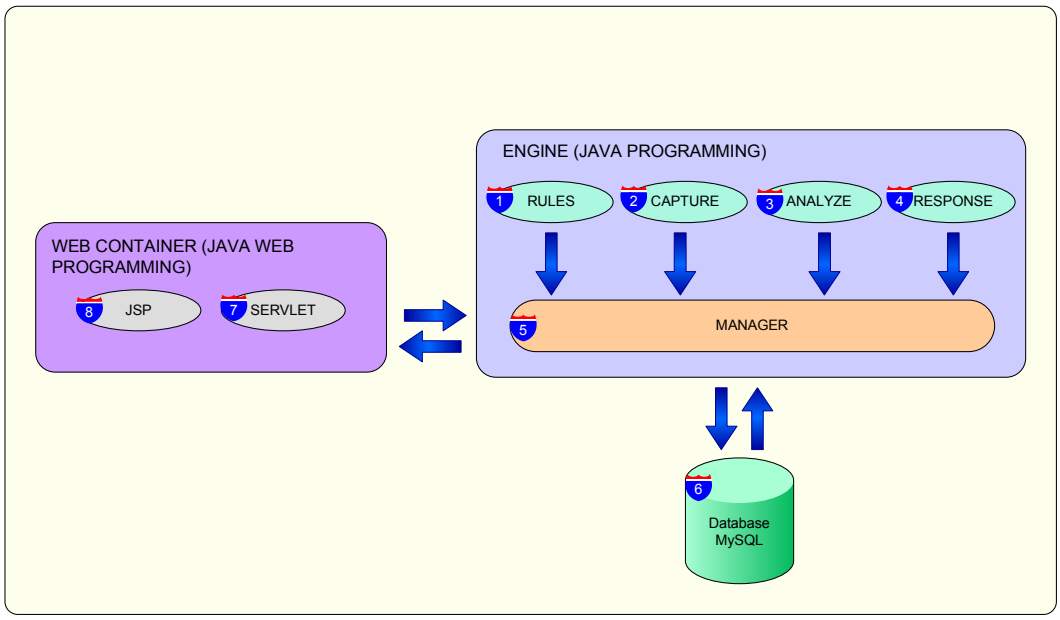
IMPLEMENTASI DAN PENGUJIAN

Tahapan lanjutan setelah analisa dan perancangan ialah tahapan implementasi dan pengujian.

5.1 Implementasi Perangkat

Tahapan implementasi ialah tahapan yang dicapai ketika perangkat telah selesai untuk digunakan dan bermanfaat untuk melihat ketercapaian sistem yang dibangun sesuai dengan tujuan pembangunan sistem. Tahapan ini terdiri dari proses penulisan program, uji sub sistem dan penggabungan sistem.

Implementasi perangkat menggunakan bahasa pemograman *Java* dan *Java Server Pages (JSP)* serta database *MySQL*.



Gambar 5.1 Implementasi Perangkat

Pada proses implementasi terdapat tiga bagian utama dalam pengembangan perangkat ini. Berikut deskripsi implementasi perangkat yang telah dilaksanakan, yaitu:

A. Implementasi *Engine,* ialah tahapan inti dari pengembangan perangkat. pada tahap ini terdapat beberapa sub bagian, sebagai berikut:

1. *Rules* merupakan implementasi dari *class* data-data inputan, misalnya : *Signature Host*, *Signature Network*, Kondisi Normal *Host* dan Kondisi Normal *Network*
2. *Capture* ialah implementasi data yang dihasilkan ketika proses *capture* (penangkapan) kondisi *network* dan *host*.
3. *Analyze* adalah implementasi dari data proses pendeteksian terhadap penyerangan
4. *Response* merupakan implementasi data respon yang telah dilakukan terhadap penyerangan
5. *Manager* ialah penghubung antara *class rules, capture, analyze* dan *response* dengan pemrograman berbasiskan *web*, serta berperan dalam pengaturan method-method yang akan dieksekusi terhadap *class.*

B. Implementasi Berbasiskan *web*

6. *Servlet* adalah penghubung dinamis antara *web server* dengan *user.* Atau disebut sebagai penghubung tampilan *Java Server Pages* dengan *web server*. *Servlet* berjalan disisi *server.*
7. *Java Server Pages (JSP)* merupakan halaman web yang berisi tampilan yang dapat dilihat secara kasat mata oleh *user*. Setiap eksekusi yang

akan dilakukan pada perangkat oleh *user*, selalu dimulai dengan interaksi dengan halaman *JSP* ini

C. Implementasi *Database*

8. *Database* yang digunakan ialah *MySQL*

5.1.1. Lingkungan Implementasi

Lingkungan implementasi yang digunakan terdiri dari dua lingkungan perangkat yaitu :

1. Perangkat Keras

   Perangkat keras yang digunakan memiliki spesifikasi sebagai berikut:

   a. *Processor* : Intel Core 2 Duo

   b. *Memory* : 1 GHz DDR

   c. *Hardisk* : 180 GB

2. Perangkat Lunak

   Perangkat lunak yang digunakan ialah:

   1. Sistem Operasi : *Linux* (*Fedora 10 Distribution)*
   2. Bahasa Pemrograman : *Java* dan *Java Server Pages (JSP)*
   3. DBMS : *Database* MySQL
   4. *Development Tool* : *Netbeans 6.4* menggunakan teknologi *Web Server build-in GlassfishV2*
   5. *Browser* : *Mozilla Firefox 5.3.2*

5.1.2. Batasan Implementasi

Tugas Akhir ini memiliki batasan implementasi sebagai berikut:

1. Menggunakan bahasa pemrograman *Java* sebagai *engine* pemrosesan unjuk kerja sistem dan *Java Server Pages (JSP)* sebagai tampilan antar muka sistem, serta dibangun dengan konsep *open source* dan *object oriented* yang berbasiskan web.
2. Tahap implementasi menggunakan tool *Netbeans 6.4* menggunakan teknologi *Web Server build-in GlassfishV2*
3. *Host Resource* yang menjadi objek pendeteksian adalah *web server*

5.1.3. Hasil Implementasi

Hasil implementasi perangkat ini berupa tampilan *web base Hybrid Intrusion Detection Management System* (*HydManSys*) yang dapat dilihat secara keseluruhan pada Lampiran.

5.2.4 Implementasi Antarmuka

Implementasi antarmuka menggunakan tampilan *web base* . Secara garis besar penjelasan sistem yang dibangun sebagai berikut :

1. Menu *Home* merupakan tampilan awal dari sistem.
2. Menu *managemt rules*. Rincian menu *management rules* sebagai berikut :

   Menu ini terbagi atas delapan bagian, yaitu :

   a. Input *Signature HIDS* berfungsi untuk melakukan penginputan data *Signature HIDS*

b. Input *Signature NIDS* berfungsi untuk melakukan penginputan data *Signature NIDS*

c. Input kondisi normal *HIDS* berfungsi untuk melakukan penginputan data kondisi normal *HIDS*

d. Input kondisi normal *NIDS* berfungsi untuk melakukan penginputan data kondisi normal *NIDS*

e. Update Signature HIDS berfungsi untuk melakukan *update* data *Signature HIDS*

f. Update Signature NIDS berfungsi untuk melakukan *update* data *Signature NIDS*

g. Update kondisi normal *HIDS* berfungsi untuk melakukan *update* data *Signature HIDS*

h. Update kondisi normal *NIDS* berfungsi untuk melakukan *update* data kondisi normal *NIDS*

3. Menu *Log.* Tampilan menu *log* terdiri atas tiga submenu yaitu:

   a. Submenu *Sniffer Mode* berfungsi untuk melakukan proses *sniffing* terhadap *traffic host* dan *network*

   b. Submenu *Logger Mode* berfungsi untuk melakukan penganalisaan kondisi *traffic host* dan *network* berdasarkan waktu *capture*

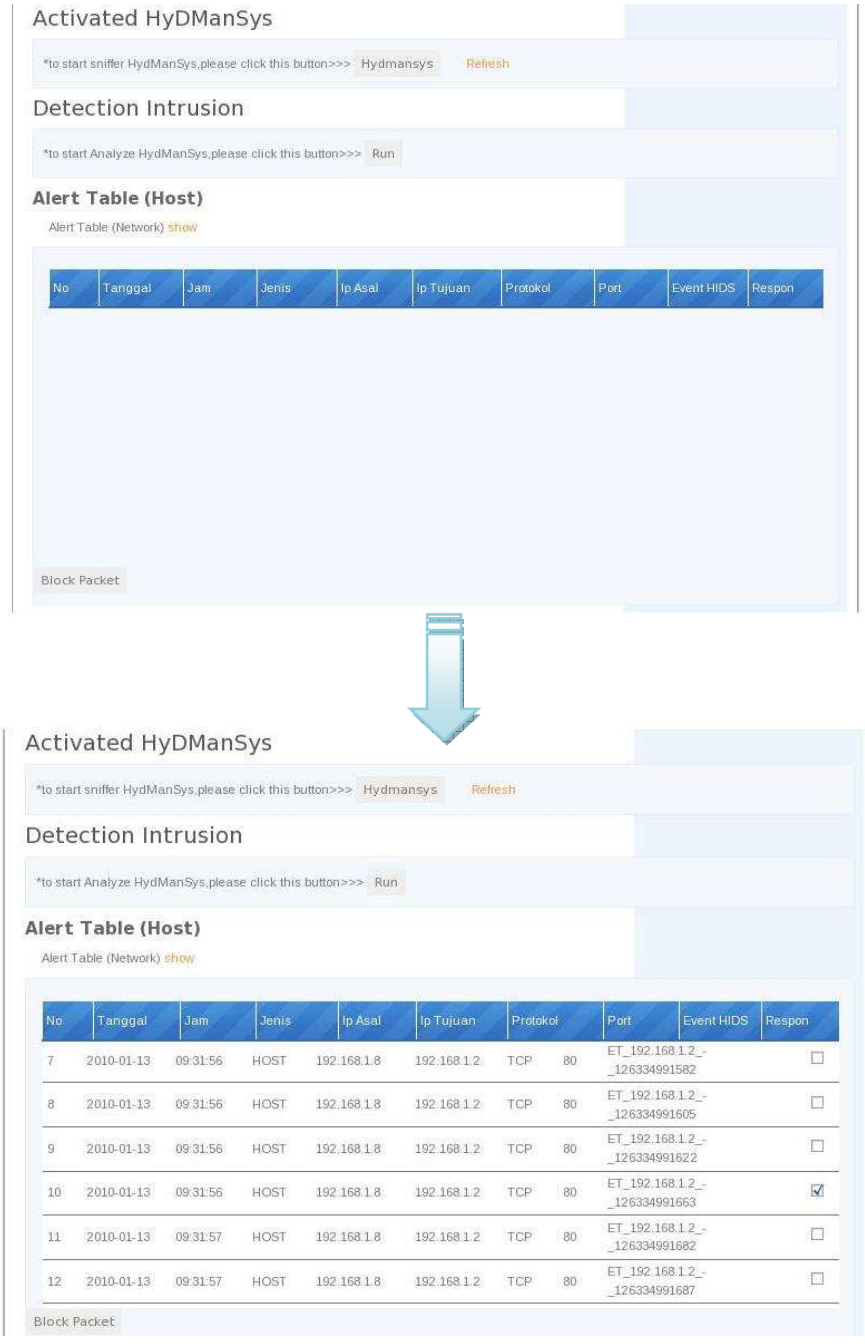
   c. Submenu *HyDManSys* berfungsi untuk melakukan proses analisa secara keseluruhan pada kondisi *traffic host* dan *network*

4. Menu *about* berfungsi untuk melihat informasi dari sistem.

5. Menu ubah *password* berfungsi untuk mengubah *password admininstaror*

6. Menu *logout* berfungsi untuk keluar dari sistem.

Dapat diketahui bahwa perangkat ini memiliki empat bagian menu yaitu *Home*, *Management Rules*, *Log* dan *About*. Tampilan unjuk kerja utama *HyDManSys* dapat dilihat dari Gambar 5.2.



Gambar 5.2 Tampilan unjuk kerja utama *HyDManSys*

Dari tampilan pada gambar 5.2 dapat dilihat unjuk kerja utama dari HyDManSys untuk pendeteksian penyusupan *host.* Pada gambaran tersebut, dapat diketahui tampilan awal dari sebuah pendeteksian penyusupan *host* hingga terdeteksinya sebuah penyusupan.

Untuk melakukan pendeteksian, aksi yang akan dilakukan yaitu melakukan pengaktifan *HyDManSys* dengan menekan tombol *HyDManSys*, memulai proses analisa penyusupan dengan menekan tombol *Run*, dan untuk melihat penyusupan yang sedang terjadi dengan melihat pada *alert table*, serta untuk melakukan *blocking access* dapat dieksekusi dengan memilih *item* data penyusupan dan menekan tombol *block packet.*

5.2 Pengujian Perangkat

Setelah tahapan implementasi selesai dilaksanakan, tahap selanjutnya yaitu pengujian perangkat.

5.2.1 Identifikasi dan Rencana Pengujian Perangkat

Identifikasi dan rencana pengujian dapat dilihat pada Tabel 5.1.

Tabel 5.1 Identifikasi dan Rencana Pengujian Perangkat

Kelas Uji	Butir Uji	Tingkat Pengujian	Jenis Pengujian	Jadwal
Link dalam sistem	Normal	Pengujian sistem HydManSys	*Black box*	15 November 2009
Percobaan penyusupan jaringan	Normal	Pengujian sistem HyDManSys	*Black box*	24 November 2009 dan 27 November 2009

5.2.1.1 Modul Pengujian *Link* Sistem

Pengujian ini bertujuan untuk melakukan evaluasi terhadap *link* sistem. Sistem ini bernama *HyDManSys (Hybrid Intrusion Detection Management System*. Tabel 5.2 menampilkan pengujian *link* sistem.

Tabel 5.2 Butir Uji Modul Pengujian *Link* Sistem

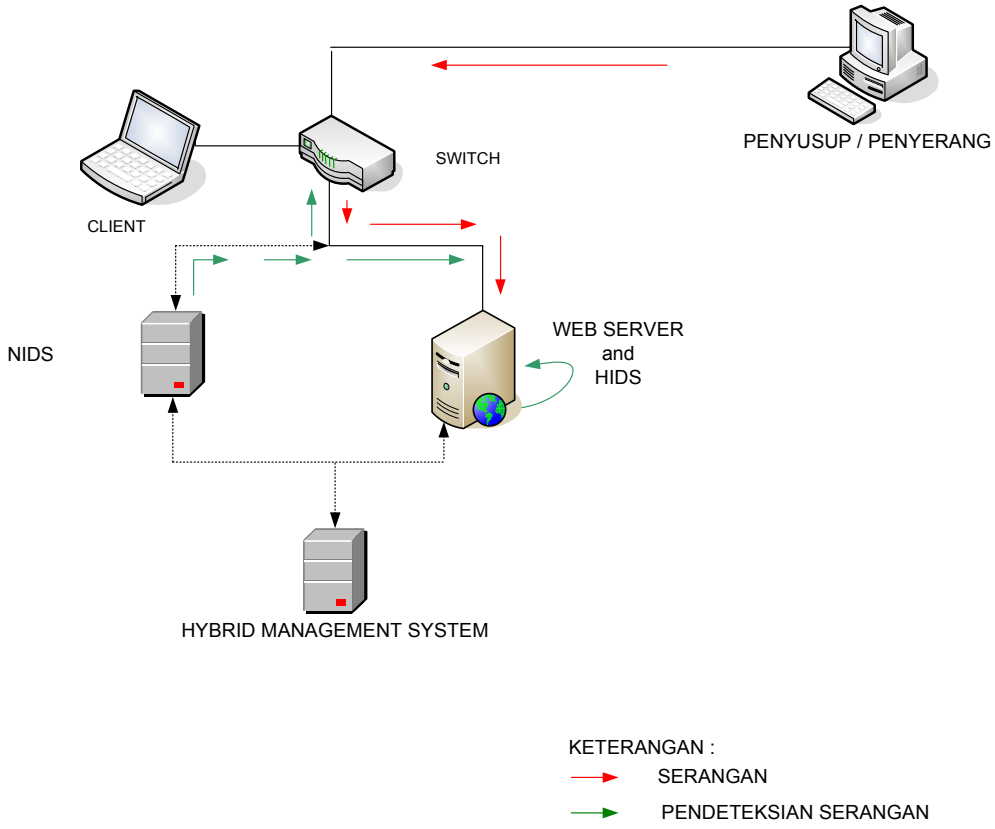
No	Deskripsi	Prekondisi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Kriteria Evaluasi Hasil	Hasil yang Didapat	Kesimpulan
1.	*Link "Login"*	Tampilan awal	Klik menu *Login*	*Username* dan *password*	Tampil halaman *Home*	Hasil sesuai dengan yang diharapkan	Tampil halaman *Home*	Berhasil
2.	*Link "Home*"	Tampilan setelah *login admin*	Klik menu *Home*	-	Tampil halaman *Home*	Hasil sesuai dengan yang diharapkan	Tampil halaman *Home*	Berhasil
3.	*Link "Management Rules "*	Tampilan setelah *login admin*	Klik	-	Tampil submenu	Hasil sesuai dengan yang diharapkan	Tampil submenu input dan update	Berhasil
4.	*Link "Log*"	Tampilan setelah *login admin*	Klik menu *Log*	-	Tampil submenu	Hasil sesuai dengan yang diharapkan	Tampil submenu *sniffer mode, logger mode,* dan *hydmansys mode*	Berhasil
5.	*Link "About*"	Tampilan setelah *login admin*	Klik menu *About*	-	Tampil informasi sistem	Hasil sesuai dengan yang diharapkan	Tampil informasi sistem	Berhasil

No	Deskripsi	Prekondisi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Kriteria Evaluasi Hasil	Hasil yang Didapat	Kesimpulan
6.	*Link* ”Ubah *Password”*	Tampilan setelah *login admin*	Klik *link* Ubah Password	*Password* baru	Jika telah berhasil diubah, maka kembali ke halaman home dan password baru dapat digunakan untuk login selanjutnya	Hasil sesuai dengan yang diharapkan	Jika telah berhasil diubah, maka kembali ke halaman home dan password baru dapat digunakan untuk login selanjutnya	Berhasil
7.	*Link* ”Logout”	Tampilan setelah *login admin*	Klik *link logout*	-	Tampil halaman *login*	Hasil sesuai dengan yang diharapkan	Tampil halaman *login*	Berhasil
8.	*Link* submenu ”Input *Signature HIDS*”	Tampilan setelah menu *Management Rules*	Klik submenu ”Input *Signature HIDS*”	*Insert* data *Signature HIDS*	Tampil data dari *database* dan daftar *Signature HIDS*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar *Signature HIDS*	Sesuai dengan *database*
9.	*Link* ”Input *Signature NIDS*”	Tampilan setelah menu *Management Rules*	Klik ”Input *Signature NIDS*”	*Insert* data *Signature NIDS*	Tampil data dari *database* dan daftar *Signature NIDS*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar *Signature NIDS*	Sesuai dengan *database*
10.	*Link* ”Input kondisi normal *HIDS*”	Tampilan setelah menu *Management Rules*	Klik ”Input kondisi normal *HIDS*”	*Insert* data kondisi normal *HIDS*	Tampil data dari *database* dan daftar kondisi normal *HIDS*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar kondisi normal *HIDS*	Sesuai dengan *database*
11.	*Link* ”Input kondisi normal *NIDS*”	Tampilan setelah menu *Management Rules*	Klik ”Input kondisi normal *NIDS*”	*Insert* data kondisi normal *NIDS*	Tampil data dari *database* dan daftar kondisi normal *NIDS*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar kondisi normal *NIDS*	Sesuai dengan *database*

No	Deskripsi	Prekondisi	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Kriteria Evaluasi Hasil	Hasil yang Didapat	Kesimpulan
12.	*Link "Update Signature HIDS"*	Tampilan setelah menu *Management Rule*	Klik *"Update* Signature *HIDS"*	*Update* data *Signature HIDS*	Tampil data dari *database* dan daftar *Signature HIDS*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar *Signature HIDS*	Sesuai dengan *database*
13.	*Link " Update Signature NIDS"*	Tampilan setelah menu *Management Rules*	Klik *" Update Signature NIDS"*	*Update* data *Signature NIDS*	Tampil data dari *database* dan daftar *Signature NIDS*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar *Signature NIDS*	Sesuai dengan *database*
14.	*Link " Update* kondisi normal *HIDS"*	Tampilan setelah menu *Management Rules*	Klik *" Update* kondisi normal *HIDS"*	*Update* data kondisi normal *HIDS*	Tampil data dari *database* dan daftar kondisi normal *HIDS*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar kondisi normal *HIDS*	Sesuai dengan *database*
15.	*Link " Update* kondisi normal *NIDS"*	Tampil setelah menu *Management Rules*	Klik *" Update* kondisi normal *NIDS"*	*Update* data kondisi normal *NIDS*	Tampil data dari *database* dan daftar kondisi normal *NIDS*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar kondisi normal *NIDS*	Sesuai dengan *database*
16	*Link " Sniffer Mode"*	Tampil setelah menu *Log*	Klik *" Sniffer Mode"*	-	Tampil data dari *database* dan daftar *sniffer host* serta *network*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar *sniffer host* serta *network*	Sesuai dengan *database*
17	*Link " Logger Mode"*	Tampil setelah menu *Log*	Klik *" Logger Mode"*	Waktu pendeteksian	Tampil data dari *database* dan daftar *logger mode*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar *logger mode*	Sesuai dengan *database*
18	*Link "HyDManSys Mode"*	Tampil setelah menu *Log*	Klik *" HyDManSys Mode"*	-	Tampil data dari *database* dan daftar analisa *HyDManSys*	Hasil sesuai dengan yang diharapkan	Tampil data dari *database* dan daftar analisa *HyDManSys*	Sesuai dengan *database*

5.2.1.2 Modul Pengujian Penyusupan Jaringan

Berikut penggambaran pengujian penyusupan jaringan yang akan dilaksanakan, pada gambar 5.3



Gambar 5.3 Pengujian Penyusupan Jaringan

Rincian penjelasan gambar pengujian, dapat dilihat pada uraian dibawah ini, sebagai berikut :

1. *Intruder* (Penyusup) yang lebih dikenal sebagai *hacker*, melakukan penyusupan pada target operasi dengan sasaran utama yaitu *Web Server* yang dimiliki sebuah *network*
2. *Hybrid IDS Management System (HyDManSys)* merupakan sistem utama yang memegang kendali terhadap keamanan traffic network. *HyIDS ManSys* memiliki beberapa tugas, yaitu:
    a. Menyimpan data kondisi *signature* dan *anomaly*
    b. Melakukan pendeteksian secara *real time* pada *resource host (Web Server)* dan *traffic* jaringan
    c. Menganalisa kondisi *traffic* jaringan dan *host* baik secara *signature* dan *anomaly detection*
    d. Memberikan peringatan (alert) terhadap kondisi yang dikategorikan sebagai penyusupan
    e. Melakukan response berupa access blocking terhadap penyusupan yang dilakukan Intruder
3. Perangkat *Web Server, HIDS, NIDS dan Hybrid Intrusion Detection Management System (HyDManSys)* secara *real* ditanamkan pada sebuah *computer server*. Gambaran diatas hanya gambaran virtual bagian-bagian yang berfungsi untuk penggambaran detail saja.
4. Runtutan peristiwa pengujian penyerangan, sebagai berikut :

a. Penyusup melakukan penyerangan terhadap jaringan lokal yang terdapat *resource (web server)* sebanyak dua kali dengan metode *Ping of Death (PoD)* dan *Denial of Service HTTP (DoSHTTP)*
b. Perangkat akan melakukan pendeteksian terhadap kondisi *network* dan host, melakukan analisa dengan metode *signature d*an *anomaly detection* terhadap kondisi tersebut, serta akan memberikan respon *blocking access* apabila *administrator* menggunakan opsi *blocking.*

Tabel 5.3 Butir Uji Modul Pengujian Penyusupan Jaringan

No	Deskripsi	Prekondisi	*Tools* Penyusupan	*IP Address* Penyusup	Prosedur Pengujian	Masukan	Keluaran yang diharapkan	Kriteria Evaluasi Hasil	Hasil yang Didapat	Kesimpulan
1.	Penyusupan *Network*	Tampilan *Hydmansys Mode*	*Script Ping of Death*	192.168.1.8 192.168.1.10 192.168.1.11 192.168.1.12 192.168.1.14 192.168.1.16 192.168.1.18 192.168.1.20 192.168.1.22 192.168.1.24	Klik *button Hydmansys, Run , refresh* serta pilih *item* yang ingin dilakukan *blocking access*	-	Tampil daftar *alert* penyusupan dan opsi *blocking* dilakukan *bocking access* terhadap penyusup	Hasil sesuai dengan yang diharapkan	Tampil *alert* penyusupan dan opsi *blocking* dilakukan *bocking access* terhadap penyusup	Berhasil
2.	Penyusupan *Host*	Tampilan *Hydmansys Mode*	*DoSHTTP*	192.168.1.8 192.168.1.10 192.168.1.11 192.168.1.12 192.168.1.14 192.168.1.16 192.168.1.18 192.168.1.20 192.168.1.22 192.168.1.24	Klik *button Hydmansys, Run, refresh,* serta pilih *item* yang ingin dilakukan *blocking access*	-	Tampil daftar *alert* penyusupan dan opsi *blocking* dilakukan *bocking access* terhadap penyusup	Hasil sesuai dengan yang diharapkan	Tampil daftar *alert* penyusupan dan opsi *blocking* serta dilakukan *bocking access* terhadap penyusup	Berhasil

5.2.2 Analisa Hasil Pengujian

Analisa hasil pengujian sistem *Hybrid Intrusion Detection Management System (HyDManSys)* dapat dilihat pada Tabel 5.9.

Tabel 5.1 Hasil Pengujian

No	Implementasi	Kelas Uji	Hasil	Deskripsi
.	*HyDManSys*	*Link* dalam sistem	Sesuai	Terhubung sesuai dengan yang diharapkan
		Percobaan penyusupan	Sesuai	Menghasilkan analisa yang sesuai dengan metode yang digunakan

5.2.3 Kesimpulan Pengujian

Hasil pengujian sistem yang telah dilakukan diperoleh kesimpulan :

1. Koneksi setiap *link* sistem dapat berjalan dengan baik
2. Penyusupan jaringan dapat dideteksi, dianalisa dan menghasilkan peringatan (*alert*) serta respon berupa *blocking access*

BAB VI

PENUTUP

Tahap akhir dari penulisan laporan ini ialah penutup, tahap ini meliputi kesimpulan dan saran yang didapat ketika telah selesai melaksanakan pengembangan perangkat secara keseluruhan.

6.1 Kesimpulan

Kesimpulan yang didapat pada pengembangan perangkat yang telah dilaksanakan, ialah *Hybrid Intrusion Detection Management System (HyDManSys)* dapat melakukan *capture*, analisa dan respon (*blocking access*) pada penyusupan jaringan sehingga berperan aktif dalam pendeteksian, dengan mengintegrasikan metode *Hybrid Intrusion Detection System, Signatures* dan *Anomaly Detection.* Namun, pada pengembangan perangkat ini, memiliki kekurangan dalam pendeteksian penyusupan, yang disebabkan oleh pembatasan masalah tugas akhir ini yaitu pada indikator yang digunakan untuk mendeteksi penyusupan.

6.2 Saran

Berdasarkan kesimpulan yang telah dipaparkan sebelumnya, saran untuk pengembangan perangkat selanjutnya, ialah :

1. Penggunaan protokol yang tidak hanya terbatas pada protokol *TCP, UDP* dan *ICMP* misalnya penggunaan protokol *SNMP (Simple Network Monitoring Protokol)*
2. Indikator yang digunakan dalam analisa penyusupan jaringan tidak hanya terbatas pada indikator *IP Address* dan *Port*

DAFTAR PUSTAKA

Ariyus, Dony. "*Sistem Pendeteksian Penyusupan"*.Yogyakarta, Andi, 2007

_____."*Berbagai Macam Serangan Terhadap Jaringan Komputer"*. *[online]Available* http://www.konche.org/details.php?id=49, diakses 6 Januari 2010

_____, *"Jaringan".[Online] Available* http://www.virologi.info/modul/ Jaringan.pdf , diakses 3 April 2009

_____,*"Java"*. *[Online] Available* http://www.ittelkom.ac.id/ library/index.php?view=article&catid=6%3Ainternet&id=32%3Ajava&option=com_content&Itemid=15, diakses 10 Maret 2009

Setiawan, Deris. *"Bab 2 TCP/IP"*. http://ilkom.unsri.ac.id/dfiles/materi/jarkom/ bab2 -TCPIP.pdf, diakses 3 April 2009